BUSTLE: A NEW CIRCUIT SIMULATION TOOL USING ASYMPTOTIC WAVEFORM EVALUATION AND PIECE-WISE LINEAR APPROACH

A THESIS
SUBMITTED TO THE DEPARTMENT OF ELECTRICRICAL AND ELECTRONICS ENGINEERING
AND THE INSTITUTE OF ENGINEERING AND SCIENCES
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

By

Cemal Tamer DİKMEN

July 1990

BUSTLE: A NEW CIRCUIT SIMULATION TOOL USING ASYMPTOTIC WAVEFORM EVALUATION AND PIECE-WISE LINEAR APPROACH

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND

ELECTRONICS ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCES

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

Cemal Tamer Dikmen

By

Cemal Tamer Dikmen

July 1990

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Dr. Abdullah Atalar(Principal Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Mehmet Ali Tan

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Ayhan Altıntaş

Approved for the Institute of Engineering and Sciences:

Prof. Dr. Mehmet Baray
Director of Institute of Engineering and Sciences

ABSTRACT

BUSTLE: A NEW CIRCUIT SIMULATION TOOL USING ASYMPTOTIC WAVEFORM EVALUATION AND PIECE-WISE LINEAR APPROACH

Cemal Tamer Dikmen
M.S. in Electrical and Electronics Engineering
Supervisor: Prof. Dr. Abdullah Atalar
July 1990

BUSTLE, a new general purpose circuit simulation program is developed especially for the analysis of VLSI circuits. BUSTLE uses Asymptotic Waveform Evaluation (AWE), which is a new method to analyze linear(ized) circuits, and PWL approach for the representation of nonlinear devices. AWE employs a form of Padé approximation rather than numerical integration to approximate the behavior of linear(ized) circuits in either the time or the frequency domain. AWE is extended to match both derivative and integral moments to overcome the unstability problem.

ÖZET

BUSTLE: ASİMPTOTİK EĞRİ TAHMİNİ VE PARÇALI DOĞRUSAL YAKLAŞIMI KULLANAN YENİ BİR DEVRE SİMÜLATÖRÜ

Cemal Tamer Dikmen
Elektrik ve Elektronik Mühendisliği Bölümü Yüksek Lisans
Tez Yöneticisi: Prof. Dr. Abdullah Atalar
Temmuz 1990

BUSTLE, özellikle VLSI devrelerinin analizinde kullanılmak için geliştirilmiş yeni ve genel amaçlı bir devre simülasyon programıdır. BUSTLE doğrusal devrelerin analizi için yeni bir metot olan asimptotik eğri tahmini yaklaşımını ve doğrusal olmayan elemanlar için parçalı doğru yaklaşımını kullanır. Asimptotik eğri tahmini doğrusal devrelerin tepkisini zaman veya frekans alanında tahmin etmek için sayısal entegral hesabı yerine bir çeşit Padé yaklaşımı kullanır. Asimptotik eğri tahmini metodunun kararsızlık problemini çözmek için türev ve entegral momentlerin birlikte eşleştirilmeleri sağlanmıştır.

ACKNOWLEDGMENT

First of all, working in a project such as this is a double pleasure for me, because it is satisfying not only to see the promising results of BUSTLE but also to work with the people in the CAD group: Prof. Abdullah Atalar, Assoc. Prof. Mehmet Ali Tan, Murat Alaybeyi and Satılmış Topçu. I am extraordinarily lucky to work with them.

I am deeply grateful to Prof. Abdullah Atalar, not only for his supervision, encouragement and invaluable advice throughout the development of this thesis, but also for creating a departmental environment so conducive to research; and to Assoc. Prof. Mehmet Ali Tan who offered extensive and helpful suggestions and provided the support of a friendly interest. I would also like to acknowledge the support of Prof. Ronald Rohrer, from Carnegie Mellon University, who started the the project following a summer course on simulation.

It is difficult to express the depth of my gratitude to Murat Alaybeyi. As a friend and a partner, he has been my main source of encouragement during the times when completing this thesis seemed to be an impossible task. I owe him a large debt. A special note of thanks is due to Satılmış Topçu for his invaluable cooperation in this work.

I am also indebted to Prof. Erol Sezer and Assoc. Prof. Ayhan Altıntaş for their crucial contributions to this project. I would like to thank Gözde Bozdağı, Gökhun Tanyer, Mustafa Karaman and M.F. Haq for their stimulating study in the early stages of the project. Many thanks also to NATO–SFS (TU-Microdesign) Project for their support.

Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

In circuit analysis and design, various circuit simulation tools are used based on different requirements of accuracy and execution speed. These requirements depend on the type and size of the circuit which is to be simulated. As expected in any numerical simulation there is a trade-off between the accuracy and speed of program execution. As the device sizes get smaller and smaller, and the number of devices in a system increases, this trade-off becomes a bottleneck in simulation of circuits, especially for VLSI circuits.

The complex nonlinear characteristic of devices and the large number of iterations needed for computing the transient response in timing simulations result in extensive computations and thus very long simulation time. Almost all of the circuit simulators use numerical and iterative methods (e.g. Newton-Raphson) to handle nonlinear characteristics and numerical integration methods (e.g. Forward Euler, Backward Euler, Trapezoidal, etc.) to compute the transient response of energy storage elements.

An important point for a simulator is that it should be adaptive to new devices resulting from the emerging technology, in order to prevent it from being obsolete in a short time. Even the user must have the capability of doing this integration.

By the motivation of the above facts, a new circuit simulation tool, BUSTLE (Bilkent University Simulation Tool for Linearized Environment) is developed. Our first aim is to terminate the simulation by success (no convergence problem). Instead of numerical integration methods, Asymptotic Waveform Evaluation (AWE) technique is employed in BUSTLE to compute the response of energy storage elements.

One of our major goals is to finish the job in a reasonably short time. To achieve this, Piece-wise-linear (PWL) approach is used to characterize the nonlinear elements. The main reason behind our choice of PWL approximation is that it deals with a set of linear equations and avoids solving of nonlinear equations. As a result of this, time complexity is decreased and the convergence in DC Analysis is guaranteed. Furthermore, AWE is mainly for linear(ized) circuits. Therefore, the use of PWL approximation makes the utilization of AWE easy and efficient for nonlinear devices.

Another important reason for choosing the PWL approximation is flexibility so that the user can easily define his own device models for nonlinear devices. Subsequently, it provides the user to make an optimal trade-off between the accuracy and the speed of the simulation. This makes the simulator independent from the technology.

Using AWE, transient analysis is the part that needs the most attention. Since we find the approximate poles and the residues for any output of the circuit, only a simple plotting routine does the AC analysis. A DC analysis is already performed prior to the transient analysis to find the operating points of the circuit. The sensitivity analysis using AWE technique may also be handled with a little additional cost [5,25].

This project was done in collaboration with M. Murat Alaybeyi and Satılmış Topçu. The work related to Asymptotic Waveform Evaluation belongs to myself. The sparse matrix solver routines [1] was written by M. Murat Alaybeyi. DC Analysis part was constructed by M. Murat Alaybeyi and myself. The work for Transient Analysis was also carried out by M. Murat Alaybeyi, Satılmış Topçu and myself. We have implemented BUSTLE, in the C Programming language on SUN-3/60 and SUN-3/110 Workstations running under the UNIX operating system.

The organization of the thesis is as follows: Chapter 2 explains the form of network equations, the solution method to solve the network equations, (LU decomposition), PWL modeling of nonlinear devices and the algorithm used in finding the operating points for the nonlinear circuits. Chapter 3 focuses on the *Asymptotic Waveform Evaluation.* It includes theoretical background for AWE and introduces the *Derivative Moment* concept and *the Moment Shifting algorithm* to handle the unstability problem of AWE. It also explains the computation of derivative and integral moments and how to handle the loops and cutsets of energy storage elements. Chapter 4 describes the method of transient analysis using AWE and PWL devices. Some implementation issues

of the transient analysis are also mentioned in this chapter. In Chapter 5, illustrative examples are provided for a variety of circuits and the simulation results are compared with SPICE. Conclusions are in Chapter 6. In addition, BUSTLE User's Guide is supplied in the Appendix part with several examples.

Chapter 2

DC ANALYSIS

Finding the "operating point" or "DC solution" of a network is usually the first step in the analysis of nonlinear networks. It involves determining the node voltages for given values of DC sources and is equivalent to the solution of nonlinear algebraic systems of equations. In this work, piece-wise linear (PWL) representation is used to characterize the nonlinear elements. Therefore, the nonlinear network is replaced by a piecewise-linear network with a corresponding simplification of the problem. Consequently, solution of nonlinear algebraic systems of equations are reduced to the solution of a set of linear systems of equations.

$$\mathbf{M}\,\mathbf{x} = \mathbf{b} \tag{2.1}$$

where $\mathbf{M}$ is the matrix that describes the resistive network, and $\mathbf{b}$ is the source vector.

2.1 Network Equations

Kirchhoff current law (KCL), Kirchhoff voltage law (KVL) and the element constitutive equations (CE) are used to describe a resistive linear network. There are three general methods to formulate network equations, *Nodal formulation, Mesh formulation and Tableau formulation* [26].

Nodal admittance formulation is based on the Kirchhoff current law which states: *The algebraic sum of currents leaving any node is zero.*

$$\mathbf{Y}\,\mathbf{V} = \mathbf{J} \tag{2.2}$$

where $\mathbf{Y}$ is the nodal admittance matrix, $\mathbf{J}$ is the current source vector, and $\mathbf{V}$ is the vector containing node voltages, to be solved for.

The mesh formulation is similar to the nodal formulation and is useful in hand calculations on simple networks. The basis for the mesh formulation is the Kirchhoff voltage law: *The sum of voltage drops around any loop is zero.*

$$\mathbf{Z}\,\mathbf{I} = \mathbf{E} \tag{2.3}$$

where $\mathbf{Z}$ is the branch impedance matrix, $\mathbf{E}$ is the voltage source vector, and $\mathbf{I}$ is the vector containing branch currents.

These two formulation methods are quite efficient and have been used successfully in many applications, but they can not handle all ideal elements. To avoid restrictions, Tableau Analysis Method is used for the formulation of network equations [26]. In Tableau Analysis Method, all equations describing the network are collected into one large matrix equation, involving the KCL, KVL, and the constitutive equations. KCL can be expressed by

$$\mathbf{A}\,\mathbf{I}_b = 0 \tag{2.4}$$

whereas the KVL is given by

$$\mathbf{V}_b - \mathbf{A}^T\,\mathbf{V}_n = 0 \tag{2.5}$$

where $\mathbf{A}$ is the *incidence matrix* [29], $\mathbf{I}_b$ is the vector containing branch currents, $\mathbf{V}_b$ and $\mathbf{V}_n$ are the branch and node voltages respectively.

Branch constitutive equations can be written as

$$\mathbf{G}\,\mathbf{V}_b + \mathbf{R}\,\mathbf{I}_b = \mathbf{w} \tag{2.6}$$

where $\mathbf{w}$ is the vector including the independent current and voltage sources, as well as the influence of initial conditions on capacitors and inductors. This vector also includes the equivalent sources due to linearization of nonlinear elements.

Equations (2.4)-(2.6) can be put into one matrix equation.

$$\underbrace{\begin{bmatrix} \mathbf{I} & 0 & -\mathbf{A}^T \\ 0 & \mathbf{A} & 0 \\ \mathbf{G} & \mathbf{R} & 0 \end{bmatrix}}_{\mathrm{M}} \underbrace{\begin{bmatrix} \mathbf{V}_b \\ \mathbf{I}_b \\ \mathbf{V}_n \end{bmatrix}}_{\mathrm{x}} = \underbrace{\begin{bmatrix} 0 \\ 0 \\ \mathbf{w} \end{bmatrix}}_{\mathrm{b}} \tag{2.7}$$

Some advantages of Sparse Tableau Analysis is that, partitioning of M matrix is straightforward in this type of formulation. And all branch voltages, branch currents and node voltages are computed by solving the equation

(2.7). But in the Nodal analysis, only the node voltages are calculated, branch voltages and branch currents should be computed separately resulting in more complicated programs.

On the other hand, the resulting matrices are always quite large in Tableau formulation. But these matrices are more sparse than the ones in the Nodal formulation. However, if a good sparse matrix solver routine is available, then the efficiency becomes better than the Nodal analysis.

2.2 LU Decomposition

In network applications, the solution of algebraic equations is best performed by the *triangular decomposition* or **LU** *factorization* technique [26]. Algorithms for triangular decomposition are closely related to Gaussian elimination, though the computations might be performed in a different sequence. The main advantage of triangular decomposition over Gaussian elimination is that it enables simple solution of systems with different right-hand-side vectors.

Let the systems of equations be given by (2.1) and assume that the matrix $\mathbf{M}$ can be factored as follows:

$$\mathbf{M} = \mathbf{L}\,\mathbf{U} \tag{2.8}$$

where $\mathbf{L}$ is a *lower triangular matrix* and $\mathbf{U}$ is a an *an upper triangular matrix.* Then the systems of equations can be rewritten as follows:

$$\mathbf{L}\,\mathbf{U}\,\mathbf{x} = \mathbf{b} \tag{2.9}$$

Define an auxiliary vector $\mathbf{z}$ as

$$\mathbf{U}\,\mathbf{x} = \mathbf{z} \tag{2.10}$$

At this time, $\mathbf{z}$ can not be calculated because $\mathbf{x}$ is unknown. However, substituting $\mathbf{z}$ into (2.9) we get

$$\mathbf{L}\,\mathbf{z} = \mathbf{b} \tag{2.11}$$

Due to the special form of $\mathbf{L}$, the vector $\mathbf{z}$ can be calculated very simply. This is called *forward elimination* or *substitution* process. Since we know $\mathbf{z}$, again equation (2.10) can be calculated very easily due to the special form of $\mathbf{U}$. This process is called *backward substitution.*

Thus, once we LU factor the M matrix, then solving equation (2.1) becomes very easy and fast for different **b** vectors by *forward and backward substitution* (FBS). So, the main task in solving the equation (2.1) is the LU factorization of the M matrix.

As we said before, the **M** matrix is very sparse which means most of the elements of the **M** matrix is zero. One can save a lot of operations by not performing the multiplications and additions on these zeros. In fact, the zeros need not even be stored, thus reducing the memory requirements.

A sparse matrix algorithm is used in BUSTLE to solve the network equations. The **M** matrix is stored in a suitable structure to make the LU factorization efficient and fast. The detailed information on the data structure and the algorithm for LU factorization is described in [1].

2.3 Modeling of Nonlinear Devices

Modeling is the process by which the electrical properties of a semiconductor device is represented by means of mathematical equations or tables. Physical device models usually involve many complicated equations. Typical timing studies have shown that the major part of the computational effort in network analysis is spent in evaluating these complicated relationships. Further, most analysis methods also require derivatives of the model equations, which is a cumbersome and error-prone task for the designer. The iterative methods, such as Newton-Raphson, to solve the nonlinear equations do not guarantee the convergence. In order to avoid these problems, piece-wise linear approach is used in BUSTLE for the modeling of nonlinear devices. It should be recognized that if a large number of sample points are selected to define a nonlinear characteristics, the description approaches that of a continuous function. But in this case the execution time may be very long. Table models are employed to describe two and three terminal nonlinear devices.

2.3.1 Modeling of two-terminal nonlinear devices

The i-v characteristics of two-terminal nonlinear devices are defined by the sample i-v values extracted from the nonlinear characteristics. BUSTLE assumes that the i-v characteristics is linear between these points. By using these values, the resistance R_l, the conductance G_l, and the equivalent source

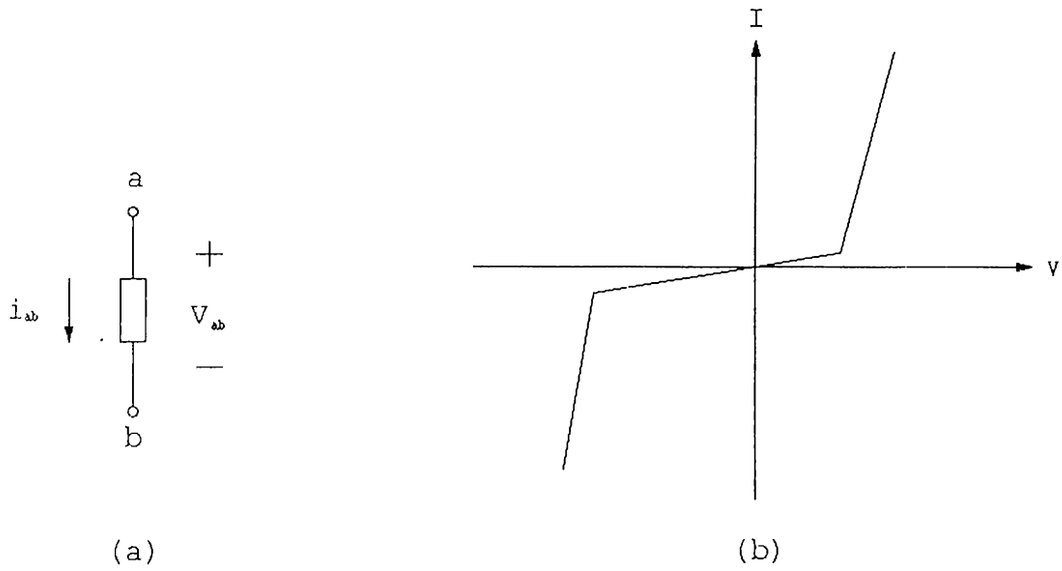

Figure 2.1: (a) Representation of a two-terminal nonlinear device; (b) i-v characteristics of a two-terminal nonlinear device.

w_l due to linearization, of a two-terminal nonlinear device can be calculated very easily.

In the tableau formulation, the nonlinear elements may be either voltage or current controlled. Both can be implemented by their piece-wise linear approximation. Note that in the l^{th} segment having the slope G_l (for the voltage-controlled case), or R_l (for the current-controlled case) there is an additional source i_l^0 or v_l^0 respectively coming from the linearization of the nonlinear devices. The equations for the element in the l^{th} segment are

$$\begin{aligned} i &= i_l^0 + G_l v \\ v &= v_l^0 + R_l i \end{aligned} \tag{2.12}$$

and, in general,

$$\mathbf{G}_l\, \mathbf{V}_b + \mathbf{R}_l\, \mathbf{I}_b = \mathbf{w}_l \tag{2.13}$$

The tableau equations can be written as follows:

$$\begin{bmatrix} \mathbf{I} & \mathbf{0} & -\mathbf{A}^T \\ \mathbf{0} & \mathbf{A} & \mathbf{0} \\ \mathbf{G}_l & \mathbf{R}_l & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{V}_b \\ \mathbf{I}_b \\ \mathbf{V}_n \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{w}_l \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{w} \end{bmatrix} \tag{2.14}$$

or in compact form

$$\mathbf{M}_l\, \mathbf{x}_l = \mathbf{w}_l + \mathbf{w} \tag{2.15}$$

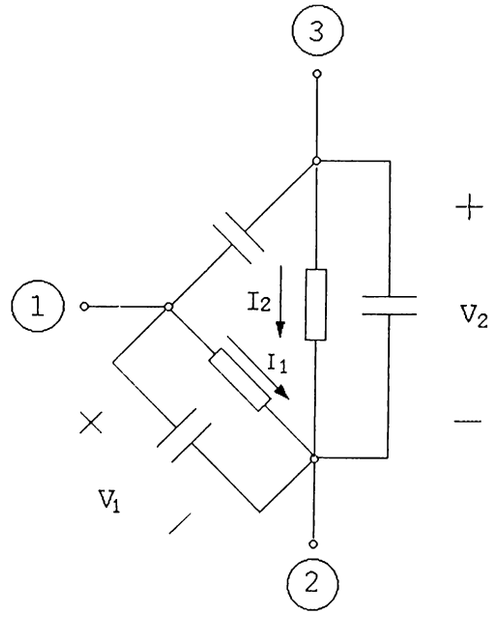

Figure 2.2: Representation of a three-terminal nonlinear device.

The subscript l denotes the *segment* in which the network operates. The right-hand-side vectors, $\mathbf{w}_l$ and $\mathbf{w}$ denote the equivalent sources due to linearization and the independent sources, respectively; they are written separately for clarity.

2.3.2 Modeling of three-terminal nonlinear devices

Three-terminal nonlinear devices are represented as a combination of two 2-terminal device placed between the three nodes as shown in Figure 2.2. There is no need to place another 2-terminal device between the nodes 1 and 3 since its voltage and current are also defined by the other two due to KVL and KCL. The parasitic capacitors are also included in the device model. The values of these capacitors are given in the model card.

The characteristics of a three-terminal nonlinear device is defined by two equations and a number of boundaries for each region. These two equations define an hyperplane in the 4-dimensional space which describes the i-v characteristics of the nonlinear device where the boundaries describe the region at which these equations are valid. The branch equations for the three-terminal nonlinear devices are of the form:

$$a_1 v_1 + a_2 v_2 + a_3 i_1 + a_4 i_2 + a_5 = 0 \tag{2.16}$$

and at most three of the coefficients a_1, a_2, a_3, a_4 can be nonzero since one of them can be eliminated using the other branch equation. The boundaries are described by the inequalities of the following form,

$$a_1 v_1 + a_2 v_2 + a_3 i_1 + a_4 i_2 + a_5 \geq 0 \tag{2.17}$$

and at most two of the coefficients a_1, a_2, a_3, a_4 can be nonzero since they have to satisfy the two branch equations, two of the variables can be eliminated. Each nonlinear device must contain a segment that satisfies the origin (both the equations and the boundaries) in order to have a valid solution when all of the independent sources are killed. This is required in order to start the DC analysis which will be described in the next section. The coefficients of the branch equations, which are called stencils, directly contribute the M matrix given in equation (2.14) and (2.15). Note that this modeling scheme does not introduce any restriction to the *user defined model* approach.

2.4 DC Analysis: Finding the Operating Point for Nonlinear Devices

The solutions to a circuit with DC inputs are called *operating points.* The term *DC Analysis* refers to the determination of operating points. For DC solution, all inductors are short-circuited and all capacitors are removed from the circuit. Given a valid solution $\mathbf{x}_0$ for an arbitrary source vector $\mathbf{y}_0$ and satisfying the boundaries of the region R_0,

$$\mathbf{M}_0 \, \mathbf{x}_0 \; = \; \mathbf{w}_0 \; + \; \mathbf{y}_0 \tag{2.18}$$

we would like to find the solution $\mathbf{x}$ and the region R_f for a given source vector $\mathbf{y}$.

$$\mathbf{M}_f \, \mathbf{x} \; = \; \mathbf{w}_f \; + \; \mathbf{y} \tag{2.19}$$

The algorithm used in the DC Analysis has been derived from the *Katzenelson's algorithm* [8,9] which guarantees the convergence in the DC Analysis [7]. The modified version of the Katzenelson's algorithm, used to find the solution $\mathbf{x}$ and the final operating region set R_f, is as follows:

1. Set $i \; = \; 0$.

2. Solve $\mathbf{x}$ from
   $\mathbf{M}_i \, \mathbf{x} \; = \; \mathbf{w}_i \; + \; \mathbf{y}$.

3. If $\mathbf{x}$ satisfies the boundaries of R_i then TERMINATE, else GOTO 4.

4. Let λ be the ratio of the distance from $\mathbf{x}_i$ to the first region boundary crossed when traversing from $\mathbf{x}_i$ to $\mathbf{x}$, to the distance from $\mathbf{x}_i$ to $\mathbf{x}$.

5. Compute $\mathbf{x}_{i+1}$ as
   $\mathbf{x}_{i+1} = \mathbf{x}_i + \lambda(\mathbf{x} - \mathbf{x}_i)$.

6. Set R_{i+1} to the neighbor region of R_i separated from it with the first crossed boundary.

7. Increment i and GOTO 2.

Note that for the first DC analysis, $\mathbf{x}_0$ and $\mathbf{y}_0$ can be selected as $\mathbf{0}$, since by definition, every nonlinear element is modeled to have a passive resistive segment satisfying the origin, and that satisfies the solution when all the sources are killed. It is obviously a poor starting point, but that is the only valid solution known initially. Afterwards solutions of the last DC analysis are chosen as $\mathbf{x}_0$, $\mathbf{y}_0$, and R_0. It is expected to do fewer calculations starting from the last solution, since it is more probable that the old solution is closer to the new one than the origin ($\mathbf{0}$).

Chapter 3

ASYMPTOTIC WAVEFORM EVALUATION

AWE is a recent technique for the approximation of linear, time-invariant circuits [2,3]. It is, in fact, a form of Padé approximation [10,11,12,14,15]. The main idea in AWE is that, the response of energy storage elements, which require the use of numerical integration methods, is approximated by using the method of moment matching and dominant pole-zero approximation. Analytic expressions for the response of energy storage elements are obtained using the AWE technique. Then these expressions are evaluated and used to solve the rest of the circuit.

3.1 Theoretical Background

AWE is explained in detail elsewhere [3,2]. We summarize it here for completeness and also to establish the slightly different notation.

AWE is most conveniently explained, in general, in terms of the differential state equations for a lumped, linear, time-invariant circuit:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}, \qquad \mathbf{x}(0) = \mathbf{x}_0 \tag{3.1}$$

where $\mathbf{x}$ is the n-dimensional state vector and $\mathbf{u}$ is the m-dimensional input vector. Such a circuit description can be found for all circuits.

This equation (3.1) has a well known solution. But for large circuits, the matrices $\mathbf{A}$ and $\mathbf{B}$ becomes so large that, it becomes practically impossible to solve this equation. Instead, a suitable approximation for x_i is sufficiently

good for our purposes, where x_i is the i^{th} state variable in $\mathbf{x}$. Our aim is to find such an approximation.

Suppose that the particular input $\mathbf{u}_p(t)$ is of the form:

$$\mathbf{u}_p(t) = \mathbf{u}_0 + \mathbf{u}_1 t, \qquad for \quad t \geq t_0, \tag{3.2}$$

where $\mathbf{u}_0$ and $\mathbf{u}_1$ are constant m-dimensional vectors. This corresponds to a step plus ramp type of input. In general, the input $\mathbf{u}_p(t)$ is not restricted to such simple signals, but rather could assume any form of input for which a particular solution can easily be obtained. The particular solution of the differential state equation (3.1) corresponding to $\mathbf{u}_p(t)$ in Eq. 3.2 is:

$$\mathbf{x}_p(t) = -\mathbf{A}^{-1}\mathbf{B}\mathbf{u}_0 - \mathbf{A}^{-2}\mathbf{B}\mathbf{u}_1 - \mathbf{A}^{-1}\mathbf{B}\mathbf{u}_1 t \tag{3.3}$$

Matrix $\mathbf{A}$ should not be singular for the particular solution to exist. This condition is equivalent to specifying that the circuit has a unique and well-defined DC solution (when all of the capacitors are open-circuited and inductors short-circuited, the M matrix in equation (2.7) must be non-singular). Finding a particular solution to a more general and complex input such as:

$$u_p(t) = e^{-\alpha t}\sin(\omega_p t + \phi) \tag{3.4}$$

is only algebraic manipulation.

That leaves us with the task of obtaining the homogeneous solution

$$\dot{\mathbf{x}}_h = \mathbf{A}\mathbf{x}_h \tag{3.5}$$

for which we must alter the initial conditions to compensate for those of the particular solution. The initial condition for the homogeneous equation (3.5) is:

$$\mathbf{x}_h(0) = \mathbf{x}_0 + \mathbf{A}^{-1}\mathbf{B}\mathbf{u}_0 + \mathbf{A}^{-2}\mathbf{B}\mathbf{u}_1 \tag{3.6}$$

where $\mathbf{x}_0$ is the initial conditions at time zero. One way of solving the homogeneous equation (3.5) is to use the Laplace Transform.

$$\mathbf{X}_h(s) = (s\mathbf{I} - \mathbf{A})^{-1}\,\mathbf{x}_h(0) \tag{3.7}$$

If the Laplace Transform of the homogeneous equation (3.7) is expanded using the Taylor Series expansion around $s = 0$, we get:

$$\mathbf{X}_h(s) = -\mathbf{A}^{-1}\left(\mathbf{I} + s\mathbf{A}^{-1} + s^2\mathbf{A}^{-2} + s^3\mathbf{A}^{-3} + \cdots\right)\mathbf{x}_h(0) \tag{3.8}$$

Using the definition of Laplace transform and the Taylor Series expansion around $s = 0$,

$$\begin{aligned} \mathbf{X}_h(s) &= \int_0^\infty e^{-st}\mathbf{x}_h(t)dt \\ &= \sum_{k=0}^{\infty} s^k \frac{(-1)^k}{k!} \int_0^\infty t^k \mathbf{x}_h(t)dt \qquad (3.9) \\ &\equiv \sum_{k=0}^{\infty} s^k \mathbf{m}_{-k-1} \qquad (3.10) \end{aligned}$$

where $\mathbf{m}_{-k}$ is the k^{th} moment [1] and particularly called the k^{th} integral moment. Now, if the two series in the equations (3.8) and (3.10) are matched term by term, we get the relationships,

$$\begin{aligned} \mathbf{m}_{-1} &= -\mathbf{A}^{-1}\mathbf{x}_h(0) \\ \mathbf{m}_{-2} &= -\mathbf{A}^{-2}\mathbf{x}_h(0) = \mathbf{A}^{-1}\mathbf{m}_{-1} \\ \mathbf{m}_{-3} &= -\mathbf{A}^{-3}\mathbf{x}_h(0) = \mathbf{A}^{-1}\mathbf{m}_{-2} \end{aligned}$$

$$\mathbf{m}_{-2q+1} = -\mathbf{A}^{-2q+1}\mathbf{x}_h(0) = \mathbf{A}^{-1}\mathbf{m}_{-2q+2} \qquad (3.11)$$

And the initial condition may be represented as the zero-th moment [2] , $\mathbf{m}_0$, to establish a recursive relationship.

$$\mathbf{m}_0 \equiv -\mathbf{x}_h(0) \qquad (3.12)$$

Then we can summarize the equations in (3.11) by the simple recursive relationship

$$\mathbf{m}_{k-1} = \mathbf{A}^{-1}\mathbf{m}_k, \qquad for \quad k \leq 0 \qquad (3.13)$$

Let us pick, in particular, the i^{th} state variable and focus on the i^{th} component of all moment vectors. We can make a q^{th} order ($q < n$, typically $q \ll n$, where n is the order of the circuit) transient approximation to the solution of the i^{th} state variable by matching their $2q$ moments coming from the equations (3.11) and (3.12).

$$x_{a_i}(t) = \sum_{l=1}^{q} k_l e^{p_l t} \qquad (3.14)$$

---

[1] Note that, our notation is slightly different from the notation used in [2,3]. If the equation (3.10) is rewritten according to their notation, it becomes

$$\mathbf{X}_h(s) = \sum_{k=0}^{\infty} s^k \mathbf{m}_k.$$

The reason for the different notation is to combine the derivative moments, which will be explained in the next section, with these moments easily.

[2] According to the notation of [2,3], initial condition is represented by the $(-1)^{th}$ moment, $\mathbf{m}_{-1}$.

In equation (3.14), p_l's and k_l's are the complex approximating poles and their corresponding residues, respectively. There are $2q$ unknowns in Eq. 3.14, where q of them are the poles and the remaining q are their corresponding residues. We should be able to solve them from the $2q$ equations, which comes from the moments.

If we take the Laplace Transform of (3.14) and use the Taylor Series expansion around $s = 0$ again, we get:

$$\begin{aligned} X_{a_i}(s) &= \sum_{l=1}^{q} \frac{k_l}{s - p_l} = -\sum_{l=1}^{q} \frac{k_l/p_l}{1 - s/p_l} \\ &= -\sum_{l=1}^{q} \frac{k_l}{p_l} \left(1 + \frac{s}{p_l} + \frac{s^2}{p_l^2} + \frac{s^3}{p_l^3} + \cdots\right) \end{aligned} \tag{3.15}$$

Again, if the coefficients of the s terms in the equations (3.10) and (3.15) are matched, we get the following set of equations for the i^{th} state variable upon inclusion of the initial conditions

$$\begin{aligned} -(k_1 + k_2 + k_3 + \cdots + k_q) &= [\mathbf{m}_0]_i \\ -\left(\frac{k_1}{p_1} + \frac{k_2}{p_2} + \frac{k_3}{p_3} + \cdots + \frac{k_q}{p_q}\right) &= [\mathbf{m}_{-1}]_i \\ -\left(\frac{k_1}{p_1^2} + \frac{k_2}{p_2^2} + \frac{k_3}{p_3^2} + \cdots + \frac{k_q}{p_q^2}\right) &= [\mathbf{m}_{-2}]_i \end{aligned}$$

$$-\left(\frac{k_1}{p_1^{2q-1}} + \frac{k_2}{p_2^{2q-1}} + \frac{k_3}{p_3^{2q-1}} + \cdots + \frac{k_q}{p_q^{2q-1}}\right) = [\mathbf{m}_{-2q+1}]_i \tag{3.16}$$

where $[\mathbf{m}_{-k}]_i$'s are the k^{th} moments corresponding to i^{th} state variable. Under the assumption that the moments can be calculated easily, without much of a computational cost, we have $2q$ equations. We merely need to solve $2q$ unknowns, which are the approximating poles and their corresponding residues. Unfortunately, these $2q$ equations (3.16) are nonlinear. One can attempt to solve these equations using iterative methods such as Newton-Raphson. Instead, we will try to reformulate the problem to allow for the direct solution of the approximating poles and residues. If we summarize the set of nonlinear equations given in (3.16) in matrix form such as

$$-\mathbf{V}\mathbf{k} = [\mathbf{m}_l]_i \tag{3.17}$$

$$-\mathbf{V}\mathbf{P}^{-q}\mathbf{k} = [\mathbf{m}_h]_i \tag{3.18}$$

where $[\mathbf{m}_l]_i$ and $[\mathbf{m}_h]_i$ are the low-order and high-order moments of the i^{th} state variable, respectively. $\mathbf{P}$ is a diagonal matrix that contains the poles at

the diagonals and $\mathbf{V}$ is the well-known Vandermonde Matrix [21,22] and $\mathbf{k}$ is the vector of residues corresponding to i^{th} state variable.

$$[\mathbf{m}_l]_i = \left[\begin{array}{cccc} [\mathbf{m}_0]_i & [\mathbf{m}_{-1}]_i & [\mathbf{m}_{-2}]_i & \cdots & [\mathbf{m}_{-q+1}]_i \end{array} \right]^T \tag{3.19}$$

$$[\mathbf{m}_h]_i = \left[\begin{array}{cccc} [\mathbf{m}_{-q}]_i & [\mathbf{m}_{-q-1}]_i & [\mathbf{m}_{-q-2}]_i & [\mathbf{m}_{-2q+1}]_i \end{array} \right]^T \tag{3.20}$$

$$\mathbf{V} = \left[\begin{array}{ccc} 1 & 1 & 1 \\ p_1^{-1} & p_2^{-1} & p_q^{-1} \\ \vdots & \vdots & \vdots \\ p_1^{-q+1} & p_2^{-q+1} & p_q^{-q+1} \end{array} \right] \tag{3.21}$$

$$\mathbf{k} = \left[\begin{array}{ccc} k_1 & k_2 & k_q \end{array} \right]^T \tag{3.22}$$

If the equations (3.17) and (3.18) are rewritten, we get the following relationships.

$$\mathbf{k} = -\mathbf{V}^{-1}\mathbf{m}_l \tag{3.23}$$

and

$$\mathbf{V}\mathbf{P}^{-q}\mathbf{V}^{-1}\mathbf{m}_l = \mathbf{m}_h \tag{3.24}$$

Since the Vandermonde Matrix is a modal matrix for a system in companion form [26], equation (3.24) is equivalent to

$$\mathbf{A}_c^{-q}\mathbf{m}_l = \mathbf{m}_h \tag{3.25}$$

where

$$\mathbf{A}_c = \left[\begin{array}{cccc} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ -a_0 & -a_1 & -a_2 & \cdots & -a_{q-1} \end{array} \right] \tag{3.26}$$

The characteristic equation that goes with the original q^{th} order homogeneous equation is

$$a_0 + a_1\lambda + a_2\lambda^2 + \cdots + a_{q-1}\lambda^{q-1} + \lambda^q = 0 \tag{3.27}$$

If $\mathbf{A}_c$ is recursively applied to $\mathbf{m}_l$, then equation (3.25) can be rewritten to yield the following result, [3].

$$\left[\begin{array}{ccc} m_0 & m_{-1} & m_{-q+1} \\ m_{-1} & m_{-2} & m_{-q} \\ & \vdots & \vdots \\ m_{-q+1} & m_{-q} & m_{-2q+2} \end{array} \right] \left[\begin{array}{c} a_0 \\ a_1 \\ \vdots \\ a_{q-1} \end{array} \right] = - \left[\begin{array}{c} m_{-q} \\ m_{-q-1} \\ \vdots \\ m_{-2q+1} \end{array} \right] \tag{3.28}$$

where m_{-j} is the j^{th} integral moment of the i^{th} state variable. If the above equation (3.28) is solved for a_k's, then we can form the polynomial in equation (3.29).

$$a_0 p^q + a_1 p^{q-1} + a_2 p^{q-2} + \cdots + a_{q-1} p + 1 = 0 \tag{3.29}$$

The roots of the q'th order polynomial above are the poles of the i^{th} state variable. And the corresponding residues can be computed using the equation (3.23).

Now, we know the approximating poles and the residues of the i^{th} state variable, then response of that state variable can be calculated using equation (3.14). Note that, if the approximating poles are not distinct, then the Vandermonde Matrix becomes singular and the equation (3.23) has no unique solution. In this case, the Vandermonde Matrix has the form

$$\mathbf{V} = \begin{bmatrix} 1 & 0 & 0 \\ p^{-1} & -p^{-2} & 2p^{-3} \\ p^{-2} & -2p^{-3} & 6p^{-4} \\ & \vdots & \vdots \\ p^{-q+1} & (-q+1)p^{-q} & (-q)(-q+1)p^{-q-1} \end{bmatrix} \tag{3.30}$$

As it is seen, second column is the derivative of the first one, and the third column is the derivative of the second one, etc.

If we summarize, determining the set of q approximating poles and residues, which are the dominant poles and their corresponding residues, requires: solving a q^{th} order linear equations (3.28), then solving the roots of the characteristic polynomial (3.29) which is, again, q^{th} order.

3.2 AWE Using Derivative Moments

We have presented the moment concept in the previous section. Remember that the k^{th} moment is the integral of $t^k \mathbf{x}_h(t)$ from $t = 0$ to ∞ (3.9). From now on, we will call the moments given in terms of the equation (3.9) and (3.10) as *integral moments*. Integral moments give information about the integral of the actual response. They correspond to the coefficients of the Taylor series expansion of the Laplace Transform of the original response around $s = 0$. Although AWE can find good approximations using integral moments, in some cases, we need the derivative of the actual response also. This necessity arises from the fact that, AWE may not find stable approximations which

will be explained in the next section. In these cases, derivative information may be used to overcome this difficulty. Derivative moments can be described similar to the integral ones by using the recursive relationship.

$$\begin{aligned} \mathbf{m}_1 &= -\mathbf{A}\mathbf{x}_h(0) = \mathbf{A}\mathbf{m}_0 \\ \mathbf{m}_2 &= \mathbf{A}\mathbf{m}_1 \\ \mathbf{m}_3 &= \mathbf{A}\mathbf{m}_2 \end{aligned}$$

$$\mathbf{m}_{2q-1} = \mathbf{A}\mathbf{m}_{2q-2}. \tag{3.31}$$

where $\mathbf{x}_h(0)$ is again given by the equation (3.6).

The approximating poles and their corresponding residues can be calculated using the same method described in the previous section. If we summarize, the coefficients of the characteristic polynomial are calculated from equation (3.32).

$$\begin{bmatrix} m_0 & m_1 & & m_{q-1} \\ m_1 & m_2 & & m_q \\ \vdots & \vdots & & \vdots \\ m_{q-1} & m_q & \cdots & m_{2q-2} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{q-1} \end{bmatrix} = - \begin{bmatrix} m_q \\ m_{q+1} \\ \vdots \\ m_{2q-1} \end{bmatrix} \tag{3.32}$$

where m_j is the j^{th} derivative moment of the i^{th} state variable. And the approximating poles are the reciprocals of the roots of the polynomial given in equation (3.33).

$$a_0 p^q + a_1 p^{q-1} + a_2 p^{q-2} + \cdots + a_{q-1} p + 1 = 0 \tag{3.33}$$

The corresponding residues are calculated using the equation (3.23), where $\mathbf{V}$ and $\mathbf{m}_l$ are given by

$$\mathbf{V} = \begin{bmatrix} 1 & 1 & & 1 \\ p_1 & p_2 & & p_q \\ \vdots & \vdots & & \vdots \\ p_1^{q-1} & p_2^{q-1} & & p_q^{q-1} \end{bmatrix} \tag{3.34}$$

$$[\mathbf{m}_l]_i = \begin{bmatrix} [\mathbf{m}_0]_i & [\mathbf{m}_1]_i & [\mathbf{m}_2]_i & & [\mathbf{m}_{q-1}]_i \end{bmatrix}^T \tag{3.35}$$

where p_i's are the reciprocals of the roots of the polynomial given in equation (3.33). Note that, the Vandermonde Matrix in this case is slightly different from the previous one. Also, poles are the reciprocals of the roots of the equation (3.33), instead of the simple roots of the equation (3.29). But if we

use the zeros of equation (3.33) to construct the Vandermonde matrix, then the two methods will be identical, except $1/p_i$'s should be used in equation (3.14).

Consequently, we can also perform the AWE using derivative moments. Derivative moments correspond to the coefficients of the Taylor series expansion of the Laplace Transform of the original response around $s = \infty$. Therefore, using derivative moments in AWE may give better results in transient analysis. Furthermore,using the combination of derivative and integral moments also helps us get rid of the unstability problem in AWE.

3.3 AWE Using the Combination of the Derivative and Integral Moments

Using AWE, we may find unstable approximations for stable circuits. This is because, we are trying to approximate a higher order system with a lower order one. And moments can give inconsistent information for that lower order system. For example, assume that the original waveform has a negative initial condition and goes to zero at steady state after a large positive overshoot in the initial part of the response (i.e. the area under the original response is positive). If we use the integral moments for matching, the first order approximation can not find a stable pole. Because we are trying to fit a decaying exponential, starting from a negative value (initial condition), but with a positive integral from 0 to ∞, which is obviously not possible. But, if we use the first derivative moment approximation or a second order approximation (derivative or integral or a combination of both), we can find a stable approximation. We may also end up with RHP poles using only the derivative moments. So it is a good idea to use different combinations of integral and derivative moments, for calculation of poles and residues of every state variable independently.

If we rewrite the equations for both derivative and integral moments,

$$\mathbf{m}_0 = -\mathbf{x}_h(0)$$

$$\begin{aligned} \mathbf{m}_{-1} &= \mathbf{A}^{-1}\mathbf{m}_0 & \mathbf{m}_1 &= \mathbf{A}\mathbf{m}_0 \\ \mathbf{m}_{-2} &= \mathbf{A}^{-1}\mathbf{m}_{-1} & \mathbf{m}_2 &= \mathbf{A}\mathbf{m}_1 \end{aligned}$$

$$\mathbf{m}_{-2q+1} = \mathbf{A}^{-1}\mathbf{m}_{-2q+2} \qquad \mathbf{m}_{2q-1} = \mathbf{A}\mathbf{m}_{2q-2} \tag{3.36}$$

Notation is such that, integral moments have negative indices where the derivative moments have positive, and $\mathbf{m}_0$ matches the initial conditions.

Now, we have $4q$ moments, $2q$ of them among all, are the derivative moments whereas the remaining $2q$ are integral moments. One needs $2q$ moments, including the initial condition $\mathbf{m}_0$, to solve the approximating poles and corresponding residues. The remaining $(2q-1)$ moments may contain any number of derivatives, or integrals.

We have seen two similar methods to calculate the poles and residues. One uses the integral moments, where the other uses the derivative moments. Any of the two methods may be used for the calculation of poles and residues. But it is easier to use the method which uses the integral moments when the number of integral moments used in the approximation is greater than the derivative moments, and the other method when the number of derivatives is greater.

Now, let's see the form of equations when the combination of derivative and integral moments is used.

i) When the number of integral moments > the number of derivative moments.

Let s be the number of derivative moments which is used in the q^{th} order approximation. Then the moment equation (3.28) is modified such that

$$\begin{bmatrix} m_s & m_{s-1} & m_{s-q+1} \\ m_{s-1} & m_{s-2} & m_{s-q} \\ \vdots & \vdots & \vdots \\ m_{s-q+1} & m_{s-q} & m_{s-2q+2} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{q-1} \end{bmatrix} = - \begin{bmatrix} m_{s-q} \\ m_{s-q-1} \\ \vdots \\ m_{s-2q+1} \end{bmatrix} \tag{3.37}$$

where m_j is the j^{th} integral moment if j is negative, or the j^{th} derivative moment if j is positive, for the i^{th} state variable. And the poles are the roots of the polynomial given in equation (3.29). The residues are calculated again by using the equation (3.23) where $\mathbf{V}$ and $\mathbf{m}_l$ are given by the equations (3.19) and (3.21) respectively.

ii) When the number of derivative moments > the number of integral moments.

Let again s be the number of derivative moments which is used in the q^{th} order approximation. Then the moment equation (3.32) is modified such

that

$$\begin{bmatrix} m_{-s} & m_{-s+1} & & m_{-s+q-1} \\ m_{-s+1} & m_{-s+2} & \cdots & m_{-s+q} \\ \vdots & \vdots & & \vdots \\ m_{-s+q-1} & m_{-s+q} & \cdots & m_{-s+2q-2} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{q-1} \end{bmatrix} = - \begin{bmatrix} m_{-s+q} \\ m_{-s+q+1} \\ \vdots \\ m_{-s+2q-1} \end{bmatrix} \tag{3.38}$$

where again m_j is the j^{th} integral moment if j is negative, or the j^{th} derivative moment if j is positive, for the i^{th} state variable. The poles are the reciprocals of the zeros of the polynomial given in equation (3.33). The residues are calculated by using the equation (3.23) where $\mathbf{V}$ and $\mathbf{m}_l$ are given by the equations (3.35) and (3.34) respectively.

Consequently, we can use any number of derivative moments in AWE to compute the poles and their corresponding residues by using the methods described above. This also helps us find stable approximations in AWE.

3.4 Moment Shifting Algorithm

The number of derivative moments is an important parameter in AWE. As it is mentioned before, AWE may not always find stable approximations. In such cases, one solution is to increase the order of approximation, but some examples have shown that this may not work, due to numerical inaccuracy which occurs when the order of approximation is high. The numerical inaccuracy arises from the moment matrix in equation (3.28) which becomes ill-conditioned when the approximation order increases. An interesting "rule of thumb" is given in [19] regarding the numerical limitations of Padé approximation: The highest order of denominator in a Padé approximation is about one fourth of the number of bits used in the mantissa of the computer's floating point representation. Since AWE is a form of Padé approximation, it also suffers from such limitations. Many approaches have been proposed to circumvent the unstability problem of Padé approximation [13,16,17,18]. But the computational complexity of those methods are not reasonable. However, unstability problem may be overcome by using different number of derivative moments in the approximation without increasing the order.

The algorithm used for this purpose is as follows.

- Necessary moments are computed according to the order of approximation.
- For every state variable DO

  (i) Compute the poles using the proper moments (initially start from all integral moments if the user does not redefine this parameter).

  (ii) If there is any unstable pole then
  If there are no integral moments used then
  increase the order of approximation by 1,
  else
  replace the highest order integral moment with the next derivative moment, and go to (i) to compute the poles.

  (iii) Compute the residues.

So, the number of derivative moments used in the approximation is increased one by one until a stable approximation is found or all of the integral moments are replaced with the derivative ones. If a stable approximation can not be found then the order of approximation is increased by one. And we are trying to avoid large orders, due to numerical errors and increased number of operations which consumes long time for approximation. But we have observed for a large number of examples that a stable and good approximation can be found below the 5'th order. If a stable approximation can not be found up to a certain order, (which never occurred for all the examples we tried) the order of approximation is not increased anymore, but the first derivative (Forward Euler) is used to approximate the response to shift in time. Afterwards, a new AWE is made with different initial conditions. Note that the dominant approximate poles and residues depend on the initial conditions. Also note that, using this algorithm, the order of approximation and the number of derivative moments used in the approximation may not be the same for different state variables. And it is not necessary to approximate all of the states with the same order for transient analysis as far as one can find a good approximation for that state variable.

3.5 Computation of the Derivative and Integral Moments

Moments are defined by the recursive relationship given in equation (3.36). We know that $\mathbf{m}_0$ matches the initial conditions. Therefore it is easy to

compute $\mathbf{m}_0$ first. Then, derivative and integral moments can be computed by recursively multiplying it with $\mathbf{A}$ and $\mathbf{A}^{-1}$, respectively.

In order to compute $\mathbf{m}_0$, we need to find the particular solution, $\mathbf{x}_p(t)$ in equation (3.3), corresponding to the particular input, $\mathbf{u}_p(t)$ in equation (3.2). The term $\mathbf{A}^{-1}\mathbf{B}\mathbf{u}_0$ in the $\mathbf{x}_p(t)$, is the steady-state (i.e., capacitors open circuited, inductors short circuited) solution of the circuit with the input $\mathbf{u}_0$. The term $\mathbf{A}^{-1}\mathbf{B}\mathbf{u}_1$ again corresponds to the steady-state solution of the circuit with the input $\mathbf{u}_1$. And the term $\mathbf{A}^{-2}\mathbf{B}\mathbf{u}_1$ is calculated by multiplying $\mathbf{A}^{-1}\mathbf{B}\mathbf{u}_1$ with $\mathbf{A}^{-1}$ as described in the computation of integral moments.

After the computation of $\mathbf{m}_0$, we can compute the derivative and integral moments. In order to compute the integral moments, we must multiply the $\mathbf{m}_k$ with $\mathbf{A}^{-1}$. To do this, we kill all the input sources, i.e. we set $\mathbf{u} = 0$. Assuming there are no loops of capacitors and cut-sets of inductors, replace all capacitors with current sources of value $(C \times m_k)$ and all inductors with voltage sources of value $(L \times m_k)$, where C and L are the capacitance and inductance values respectively, and m_k is the k^{th} integral moment for the corresponding state variable. Then, the voltages across the current sources (replacing the capacitors) and currents through the voltage sources (replacing the inductors) are the next integral moments.

In order to find the derivative moments, all the input sources are killed. We replace all capacitors with voltage sources and all inductors with current sources of value m_k, where m_k is the k^{th} derivative moment for the corresponding state variable. Then, the currents through the voltage sources (replacing the capacitors) and the voltages across the current sources (replacing the inductors) are $\mathbf{X}\mathbf{A}\mathbf{m}_k$, where

$$\mathbf{X} = \begin{bmatrix} \mathbf{C} & 0 \\ 0 & \mathbf{L} \end{bmatrix} \tag{3.39}$$

Then, we can compute $\mathbf{A}\mathbf{m}_k$, which equals to $\mathbf{m}_{k+1}$, by multiplying the above term by $\mathbf{X}^{-1}$.

Thus, we can compute the derivative and the integral moments by using the above recursive algorithms. Note that the computation of moments does not involve LU factorization, but a simple forward and backward substitution (FBS) for each moment.

$\mathbf{X}$ is a diagonal matrix if there are no loops of capacitors and and cutsets of inductors. Handling of capacitor loops and inductor cutsets will be described in the next section. Therefore the matrix $\mathbf{X}$ remains diagonal, thus the

multiplication by $\mathbf{X}^{-1}$ does not involve matrix inversion or LU factorization of $\mathbf{X}$ matrix but just a division by a floating point number.

3.6 Loops and Cutsets of Energy Storage Elements

In solving for the derivative moments or during the calculation of the transient response of a circuit, every capacitor is converted to a voltage source. Naturally loops of capacitors result in redundancy in the circuit equations.

BUSTLE handles this situation by first constructing a proper tree [20]. The capacitors in the tree are converted to voltage sources. The capacitors in the co-tree will form loops with tree capacitors. The voltages across the co-tree capacitors are determined by the tree capacitors. Therefore, we need to put some constraints on the currents around the loop, for a proper current distribution [4]. Let us think about a capacitor loop, with n capacitors, $C_1, C_2, \ldots, C_n$, of which C_n has been determined to be in the co-tree, and the others in the tree, it comes from the KVL that

$$v_{C_n} = v_{C_1} + v_{C_2} + \cdots + v_{C_{n-1}} \tag{3.40}$$

If the derivatives of the both sides are taken with respect to time, we get

$$\frac{dv_{C_n}}{dt} = \frac{dv_{C_1}}{dt} + \frac{dv_{C_2}}{dt} + \cdots + \frac{dv_{C_{n-1}}}{dt} \tag{3.41}$$

Equation (3.41) can be rewritten as

$$\frac{i_{C_n}}{C_n} = \frac{i_{C_1}}{C_1} + \frac{i_{C_2}}{C_2} + \cdots + \frac{i_{C_{n-1}}}{C_{n-1}} \tag{3.42}$$

Equation (3.42) defines a *current controlled current source* for C_n with the controlling branches $(C_1, C_2, \ldots, C_{n-1})$. Consequently, C_n can be replaced by a current controlled current source with the branch equation given in (3.42) which removes the redundancy in the circuit equations.

Similarly, cutsets of inductors cause a redundancy, when they are converted to current sources. The same method can also be used for inductors. In this case, the inductor in the tree will be changed to a *voltage controlled voltage source* with the branch equation

$$\frac{v_{L_n}}{L_n} = \frac{v_{L_1}}{L_1} + \frac{v_{L_2}}{L_2} + \cdots + \frac{v_{L_{n-1}}}{L_{n-1}} \tag{3.43}$$

As a result, the equations corresponding to a co-tree capacitor and a tree inductor in the circuit equations are changed, redefining them as *current controlled current sources* and *voltage controlled voltage sources*, respectively.

A similar problem exists for capacitor cutset, and inductor loops, which cause redundancies in the DC circuit equations. But this can be eliminated by using charge and flux conservations easily [6].

Chapter 4

TRANSIENT ANALYSIS

In transient analysis, BUSTLE computes the transient output variables as a function of time over a user specified time interval. In addition to the independent DC sources, any independent source can be assigned a time-dependent value for transient analysis. Ideal step changes in the time-dependent sources are also handled without loss of generality. The transient response is computed using the AWE technique.

4.1 Transient Analysis Using AWE

A DC analysis is automatically performed prior to the transient analysis with the input values at time $t = 0$, in order to determine the operating segments/regions for the PWL devices and to calculate the initial conditions of the capacitors and inductors. If an initial condition is given for a capacitor voltage or an inductor current, these elements are replaced by voltage and current sources respectively to force the initial conditions. Otherwise, inductors are shorted and capacitors are removed from the circuit. After determining the operating segments/regions for the PWL devices and initial conditions of energy storage elements, all inductors are replaced by short-circuits and all capacitors by open-circuits to calculate the steady state response, which is required for the computation of moments in AWE. The steady state response is calculated for step and ramp type of inputs separately. First, it is calculated by using only the step part of the inputs, then the slopes of the ramp inputs are used as sources to calculate the steady state response corresponding to ramp inputs. Then the response of a state variable, as a function of time,

can be written as follows.

$$x(t) = x_{ss_{step}} + m_{0,ramp} + x_{ss_{ramp}} t + \sum_{i=0}^{q} k_i e^{p_i t} \tag{4.1}$$

where $x_{ss_{step}}$ and $x_{ss_{ramp}}$ are the steady-state values corresponding to step and ramp type of inputs respectively, p_i's and k_i's are the dominant poles and their corresponding residues, respectively. $m_{0,ramp}$ for the i^{th} state variable is given as

$$m_{0,ramp} = \left[\mathbf{A}^{-1}\mathbf{x}_{ss_{ramp}}\right]_i . \tag{4.2}$$

The value of $m_{0,ramp}$ is calculated during the calculation of moments in AWE without an additional cost.

Once the dominant poles and the residues are calculated by AWE, we know the response of all energy storage elements by equation (4.1). Then all capacitors are replaced by voltage sources and all inductors by current sources of value given by the equation (4.1). By evaluating the equation (4.1) at certain time instants, we can solve the whole circuit by a simple substitution to compute all voltages and currents in the circuit.

4.2 AWE with PWL Devices in Transient Analysis

Using AWE we obtain approximate analytic expressions for capacitor voltages and inductor currents. These expressions are valid on the time axis as long as they satisfy the set of current operating regions R_i. In order to find voltages and currents of each device, these expressions are evaluated at certain time instants by using these values as sources and the circuit is solved by a mere substitution. As we progress over time the nonlinear devices in the circuit may change their segments. If occurs, we must know the time when one nonlinear device, at least, changes its segment. As soon as we realize a segment change, we go back over time and search for the time of segment change. The capacitor voltages and inductor currents for that instant of time are the initial conditions for a new DC analysis to find the new segments/regions for PWL devices. Then a new AWE is performed using the new segments/regions and the new initial conditions.

The same thing happens when there is an input change at time t_0. We evaluate the approximate expressions found for energy storage elements and solve the circuit at time t_0^- by a mere substitution. A new DC analysis is performed at time t_0^+ using the new source vector, and a new AWE is

carried out for $t \geq t_0$. For DC analysis, we can use the previous solution and segments (instead of 0 vector) as the initial valid solution. This saves a lot of computation in DC analysis.

One problem about this algorithm is that, there is an inherent instability problem near the corners of the PWL elements, if the operating point is just on the corner. Although the algorithm is guaranteed to be convergent, the oscillations around the corner point causes the simulation not to end in a feasible time. Even this situation occurs very rarely and in some specific examples, they are determined by some additional checks in the software. If such a situation is realized, then we continue the analysis by assuming one of the segments for a while. This does not cause a large error since we are already near the corner.

4.3 Calculation of the Time-step in Transient Analysis

As it has been mentioned before, we progress over time by solving the circuit at certain time instants. The selection of the time step in transient analysis is a quite critical issue for the efficiency standpoint. If the time step is chosen too small, then too many unnecessary computations must be performed. This may even cause the simulation not to terminate in a reasonable time. Conversely too large time steps may cause large errors if there exist high frequency poles with large residues. Another drawback of the large time step is that we may skip an overshoot of the waveform which may possibly cause a segment change. Therefore, the time step used in transient analysis is dynamically calculated after each FBS. In this calculation, we consider primarily the rate of change of the most rapidly changing exponential. The rate of change at time t_0 is computed by taking the derivative of equation (4.1) with respect to time at time t_0 for all the state variables.

$$\left.\frac{dx(t)}{dt}\right|_{t=t_0} = x_{ss_{ramp}} + \sum_{i=0}^{q} k_i p_i e^{p_i t_0} \tag{4.3}$$

where k_i's and p_i's are the complex residues and poles respectively for that state variable.

As a result of dynamic selection of the time step, the simulator spends more effort when there are rapid voltage or current changes, and progresses faster in the time axis if there are slow changes. This provides an event driven feature to the simulator.

Chapter 5

RESULTS

In this chapter, we will present some results obtained by BUSTLE to demonstrate its accuracy and efficieny. The program leaves the accuracy speed trade-off to the user by giving him/her a number of options. The minimum order of approximation which determines the number of moments matched in AWE is an important parameter for the accuracy of the approximation. For example, this parameter can be selected as one for a digital CMOS circuit, but this would not be sufficient for an RLC circuit that has an oscillatory response. The minimum order and also the number of derivatives that will be matched initially can be determined by the user. There are also some other parameters that can be set by the user to improve the accuracy or the speed. Another important feature is that, user can define his own models (or use the one from the library) for nonlinear devices. This provides a capability to keep pace with the emerging technology and user can control the accuracy speed trade-off by choosing the number of segments used for modeling.

All of the simulations are carried out in SUN-3/60 Workstations. And SPICE version 2G.6 is used to compare the results of BUSTLE both from accuracy and speed standpoint.

ILLUSTRATIVE EXAMPLES

RLC underdamped circuit: Figure 5.1.
The first example is taken from [3] to demonstrate the usage of derivative and integral moments together, to get stable approximations. If we use the basic AWE method [2,3], we end up with RHP poles for C2 and L3 for a second order approximation. However, using the method described in Section 3.4, we can find stable approximations for all of the state variables.

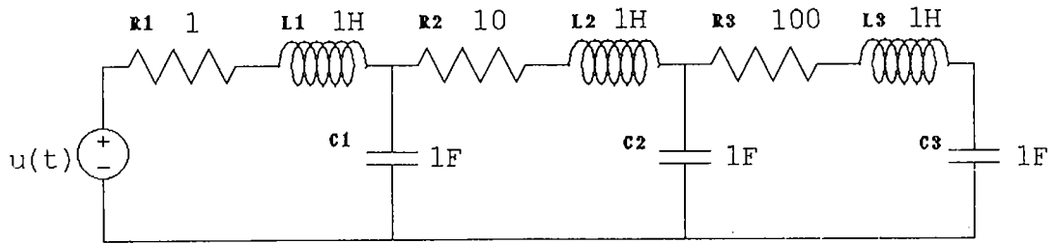

Figure 5.1: RLC underdamped circuit with real and complex poles.

2.0	2.1	3.0	3.1
-1.206e-2	-2.015e-2	**5.053e-1**	-8.935e-3
3.012e-2	-1.653e+1	-1.269e-1	-2.803e+0
		-8.915e-3	-1.965e-1

4.0	5.0	Actual
-5.556e-1 + j8.965e-1	-1.029e-1	-5.556e-1 + j8.965e-1
-5.556e-1 - j8.965e-1	-1.330e-1	-5.556e-1 - j8.965e-1
-8.914e-3	-8.914e-3	-8.915e-3
-1.023e-1	-5.556e-1 + j8.965e-1	-1.029e-1
	-5.556e-1 - j8.965e-1	-9.797e+0
		-9.998e+1

Table 5.1: Approximate Poles for response at C2 and the actual poles of the circuit.

2.0	2.1	2.2	2.3	3.0
-1.022e-2	-4.301e-9	**0.000e+0**	**0.000e+0**	-8.914e-3
4.033e-2	**4.301e-9**	-5.404e+17	-1.001e-1	-7.975e-1
				-1.047e-1

Table 5.2: Trials of BUSTLE to find a second order approximation to L3 and conclusion with a third order approximation.

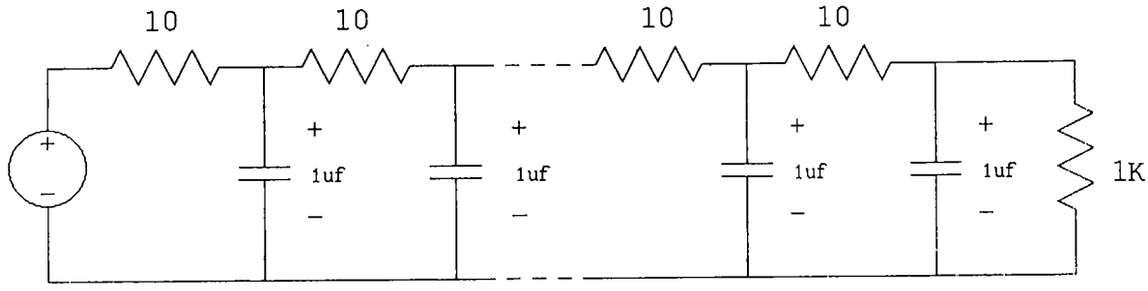

Figure 5.2: RC tree with a length of 10.

Table 5.1 and 5.2 show the approximate poles [1] of C2 and L3, respectively, for different approximation orders and different number of derivative moments used in the approximation. Notation is such that the first number in the column headers indicates the approximation order where the second indicates the number of derivative moments used in the approximation. As it can be seen from the Table 5.1 and 5.2, after using the first derivative moment, a stable approximation can be found for C2, whereas for L3 we can not get rid of RHP poles for the second order approximation even all combinations of derivative and integral moments are used. In this case the order of approximation is increased automatically, and a stable approximation in the 3'rd order is found (Table 5.2). But the remaining states are approximated in second order which is the minimum required order. So, we are doing a better approximation for L3 which improves accuracy in addition to getting rid of the unstable poles. Similarly, after using the first derivative moment, a stable approximation can be found in the third order for C2. The approximation is already stable in the fourth and fifth order without using any derivative moments. Note that, second and third order approximation are not sufficient to observe the complex poles of the response at C2. Since the RLC circuit is slightly underdamped there are two real poles closer to jw-axis than the complex conjugate pair, also the residues of the complex pair is small. With such a pole configuration, the oscillatory response at capacitor C2 is not observed until the order of approximation is 4 or greater. This is a natural consequence of the moment matching method which approximates the poles.

RC Tree with Finite Input Rise Time: Figure 5.2.
The second example is an RC tree which may be used as the model of an interconnect in a VLSI circuit. The order of the tree is 10. A transient analysis is performed using the first order approximation in AWE with a finite

[1] Numbers written in boldface indicate the unstable poles.

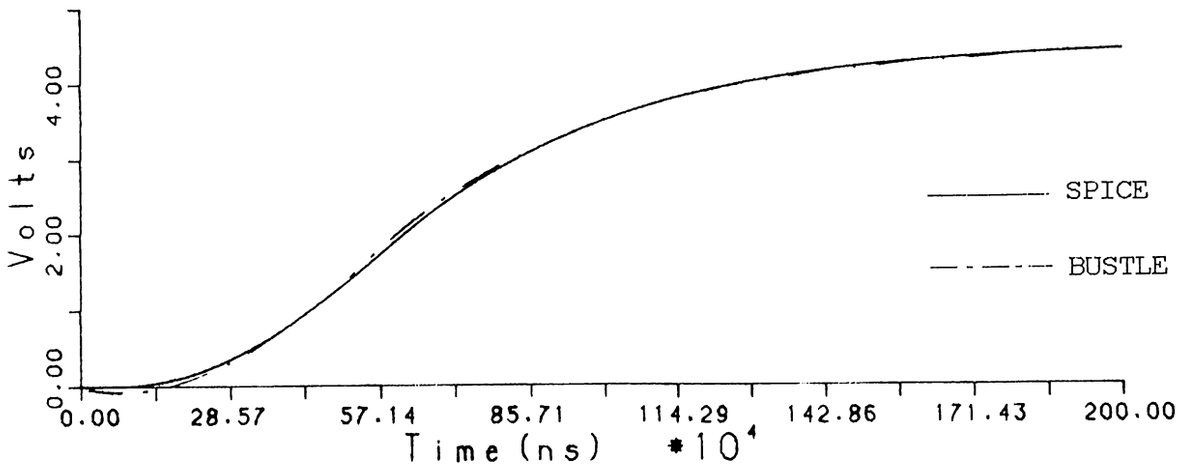

Figure 5.3: First order approximation to the response at the end of the tree.

input rise time. The simulations are performed using SPICE and BUSTLE for the response at the end of the tree (Figure 5.3). Although the first order approximation is used for a 10^{th} order circuit, the results are almost the same. The normalized RMS difference [2] between first order approximation and SPICE is 0.7%. Note that even the first order approximation is good enough for such a monotonic waveform. The required CPU time for the simulation of BUSTLE is 1.57 seconds, whereas it is 6.38 seconds for SPICE.

RLC ladder circuit: Figure 5.4.

The third example is an 14^{th} order RLC ladder. Transient analysis is performed using second and third order approximation in AWE with a 3-volt ideal step input. The results of transient analysis at the end of the ladder, performed by SPICE and BUSTLE are shown in Figure 5.5. As it can be seen from the figure, BUSTLE 3^{rd} order approximation and SPICE are indistinguishable from each other. However, the second-order AWE can not catch the oscillatory response. The normalized RMS difference between 2^{nd} order approximation and SPICE is 2.99% whereas it is 0.35% in 3^{rd} order AWE. The required CPU time for 2^{nd} order approximation of BUSTLE is 2.1 seconds where it is 2.25 seconds for the 3^{rd} order approximation. The CPU

[2]

$$Normalized\ RMS\ difference = \sqrt{\frac{\int_{t_{start}}^{t_{stop}} (x_1(t) - x_2(t))^2\, dt}{V_{max}^2\,(t_{stop} - t_{start})}}$$

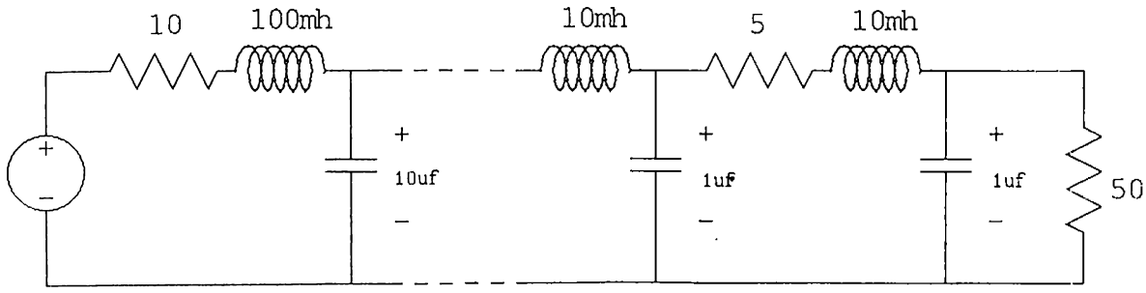

Figure 5.4: 14^{th} order RLC ladder.

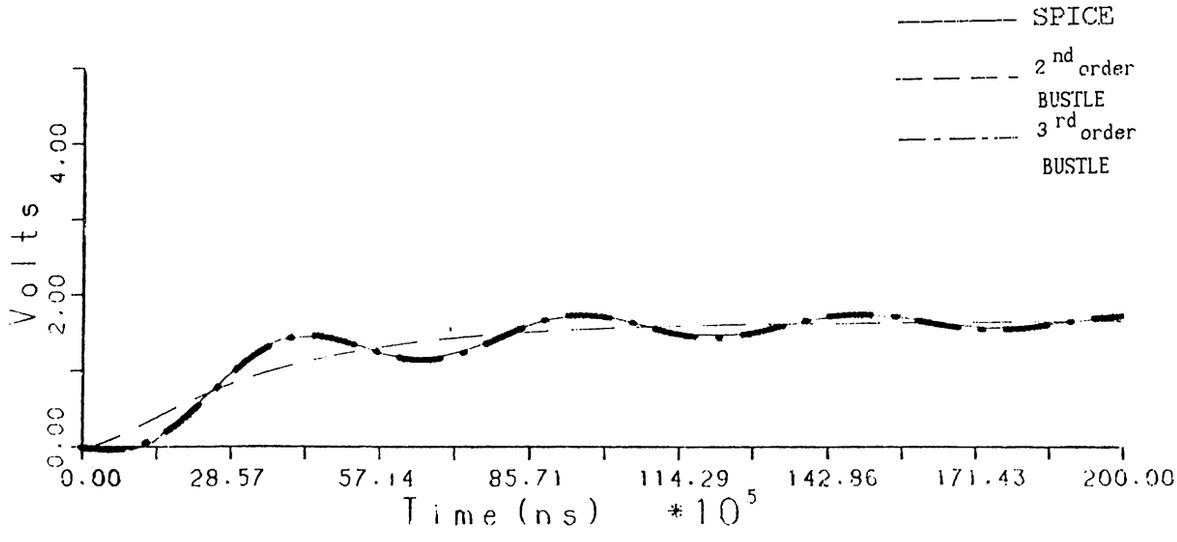

Figure 5.5: Transient simulation of the 14^{th} order RLC ladder.

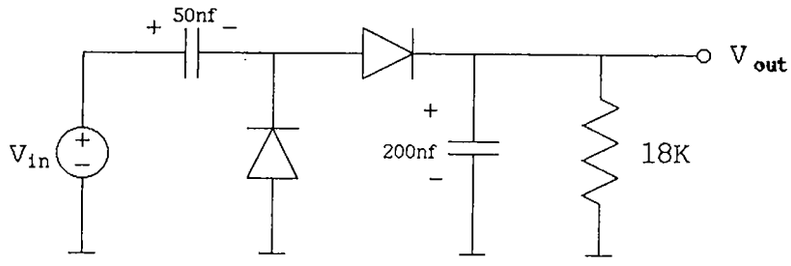

Figure 5.6: Voltage Doubler Circuit.

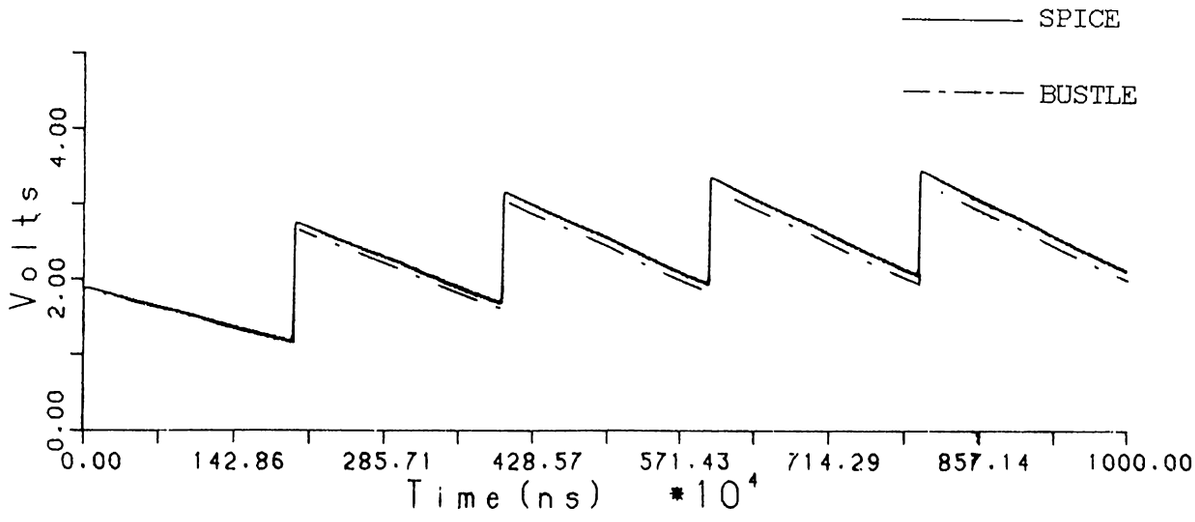

Figure 5.7: Transient analysis of the voltage doubler.

time for SPICE is 8.26 seconds.

Voltage Doubler: Figure 5.6.

The fourth example is a voltage doubler circuit with two diodes. Diodes are modeled with two segments one representing the OFF region, where the other is the ON region with $V_0 = 0.6$v. The resistance of the OFF region is $10^6\Omega$ where it is 10Ω in the ON region. Transient analysis is performed with SPICE and BUSTLE with a square wave input. The transient response at the output is observed in Figure 5.7. Although we use a very simple diode model with two segments and this circuit is very sensitive to the diode model, the results are almost the same. The normalized RMS difference between the results of BUSTLE and SPICE is 2.59%. Also note that BUSTLE can also get very good results for sharp changes in the node or branch voltages with respect to the time step where SPICE is not good in such cases. One should

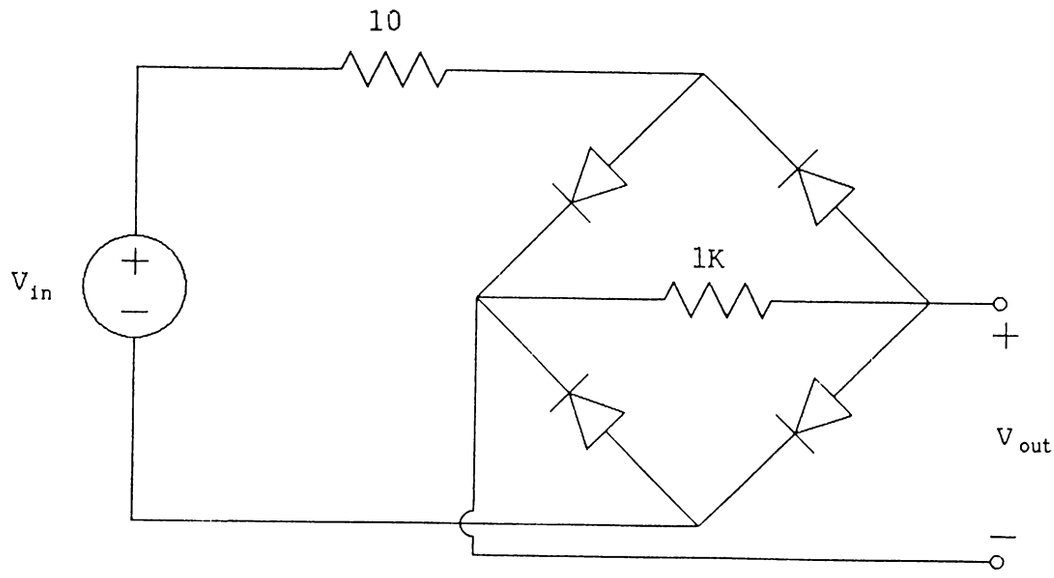

Figure 5.8: The circuit of the full-wave rectifier.

decrease the time step to a small enough value to handle such circuits in SPICE which increases the execution time. But, BUSTLE solves this problem by dynamically calculating the internal time step after each iteration. The required CPU time for transient analysis of BUSTLE is 4.0 seconds whereas it is 13.2 seconds for SPICE.

Bridge Rectifier: Figure 5.8.
The fifth example, is a full-wave rectifier with four diodes but has no energy storage element, just demonstrates that we do not lose much from the accuracy by PWL modeling. The diode model is the same as the previous example. A transient analysis is performed with SPICE and BUSTLE with a triangular wave input. The transient response at the output for both BUSTLE and SPICE is observed in Figure 5.9. The normalized RMS difference between the results of BUSTLE and SPICE is 1.67%. Note that the difference which comes from the PWL modeling of the diodes is very small between the two simulations. We also made simulations of this circuit using a diode model with 12 segments that is extracted from SPICE, the results of both simulations are almost identical. Eventually two segments are good enough to model a diode, there is no need to use a more complex model. The CPU time required for this circuit is 2.83 seconds for BUSTLE and 6.33 seconds for SPICE. The ratio of the CPU times between SPICE and BUSTLE gets larger as the circuit gets larger. By PWL modeling of diodes not only we gain speed but also the convergence in DC analysis is guaranteed using the

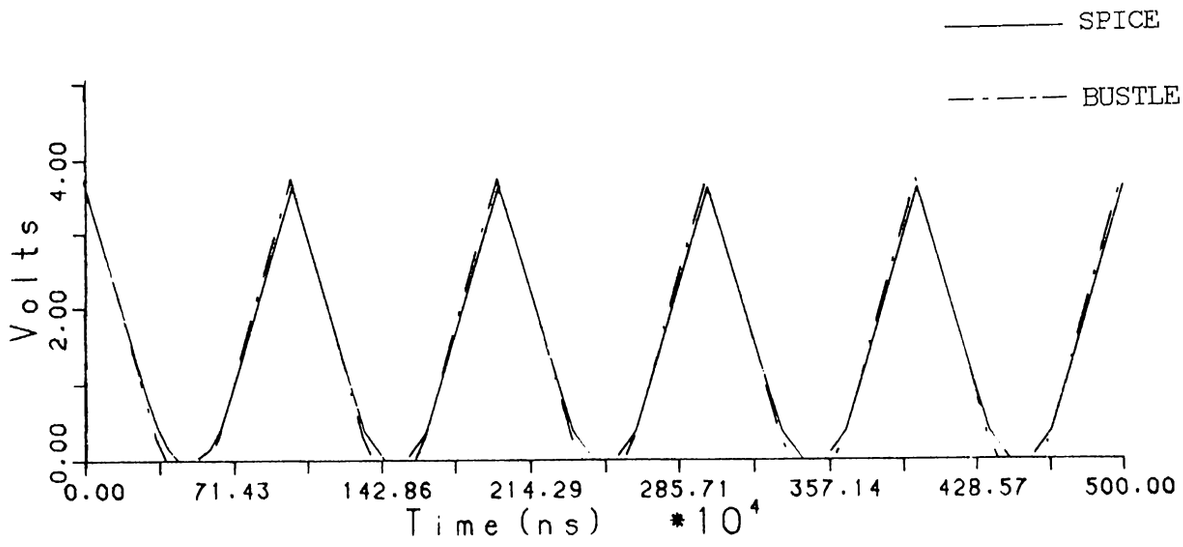

Figure 5.9: Transient analysis of the bridge rectifier.

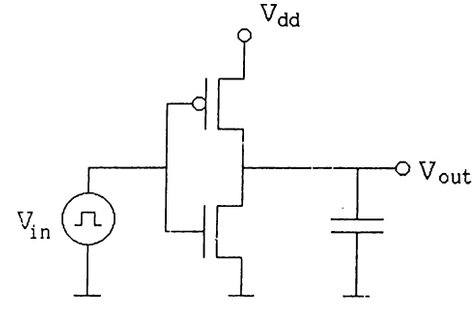

Figure 5.10: CMOS Inverter.

algorithm described in Section 3.4.

CMOS Inverter: Figure 5.10.
The sixth example is a CMOS inverter with two transistors, one is NMOS and the other is PMOS. Both type of transistors are modeled with 4 regions, cutoff, linear, saturation and inverse saturation (Figure 5.11). The inverter is loaded with a 2-pf capacitor. The input waveform contains two pulses. A transient analysis is performed with SPICE and BUSTLE and the results are given in Figure 5.12. SPICE_1 is the SPICE simulation using 50 time steps in transient analysis where SPICE_2 uses 400 time steps for the same interval. It is surprising that the two waveforms are different. We also used 50 and 400 time steps in transient analysis for BUSTLE, but our results do not differ with changing time step, which is the expected result. The simulation results of BUSTLE is very close to SPICE_2, which is expected to be more accurate, as

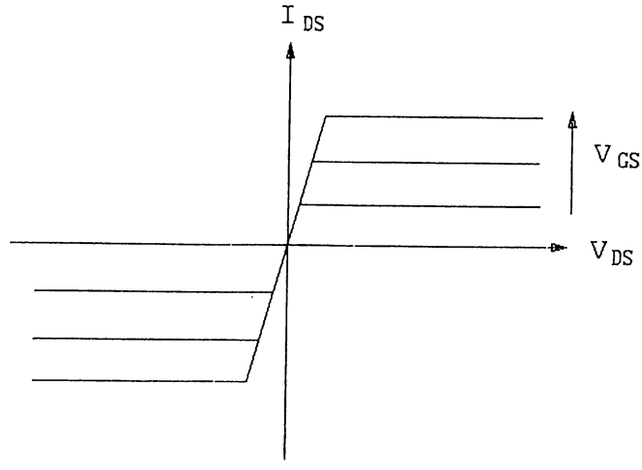

Figure 5.11: Model of the MOSFET's used in CMOS inverter.

seen from the Figure 5.12. The normalized RMS difference between BUSTLE and SPICE_2 is 2.7%, where it is 12.66% between SPICE_1 and SPICE_2. Although BUSTLE uses simple PWL models for transistors, its results are more accurate than SPICE in case of an unsuitable selection of transient time step. The selection of the transient time step does not effect the simulation results of BUSTLE, but significantly changes the results of SPICE. The CPU time required for the transient analysis of the CMOS inverter is 4.7 seconds for BUSTLE, 9.5 seconds for SPICE_1 which gives wrong results, and 11.8 seconds for SPICE_2.

A new facility of BUSTLE is that user can see the operating segments of PWL devices, which he may would like to observe, easily by adding a **SEG** command in the print card. The operating segments of the MOSFETS of the inverter can be seen also in Figure 5.12. This facility is a lot of help to the user, because it is easier to understand the PWL models which have been conventional for nonlinear elements. For example a designer generally thinks the diode as a device which is ON or OFF, instead of an exponential characteristic. We believe that BUSTLE is highly educational from this point of view since the solution style is very similar to a manual solution style.

CMOS NAND Gate: Figure 5.13.

The seventh example is a CMOS NAND gate with four transistors as given in Figure 5.13. The transistor models are the same as the previous example (Figure 5.13). The gate is loaded with a 0.5pf capacitor. A transient analysis is performed again using SPICE and BUSTLE with the inputs shown in Figure 5.14, and the results are again given in the same figure. The results

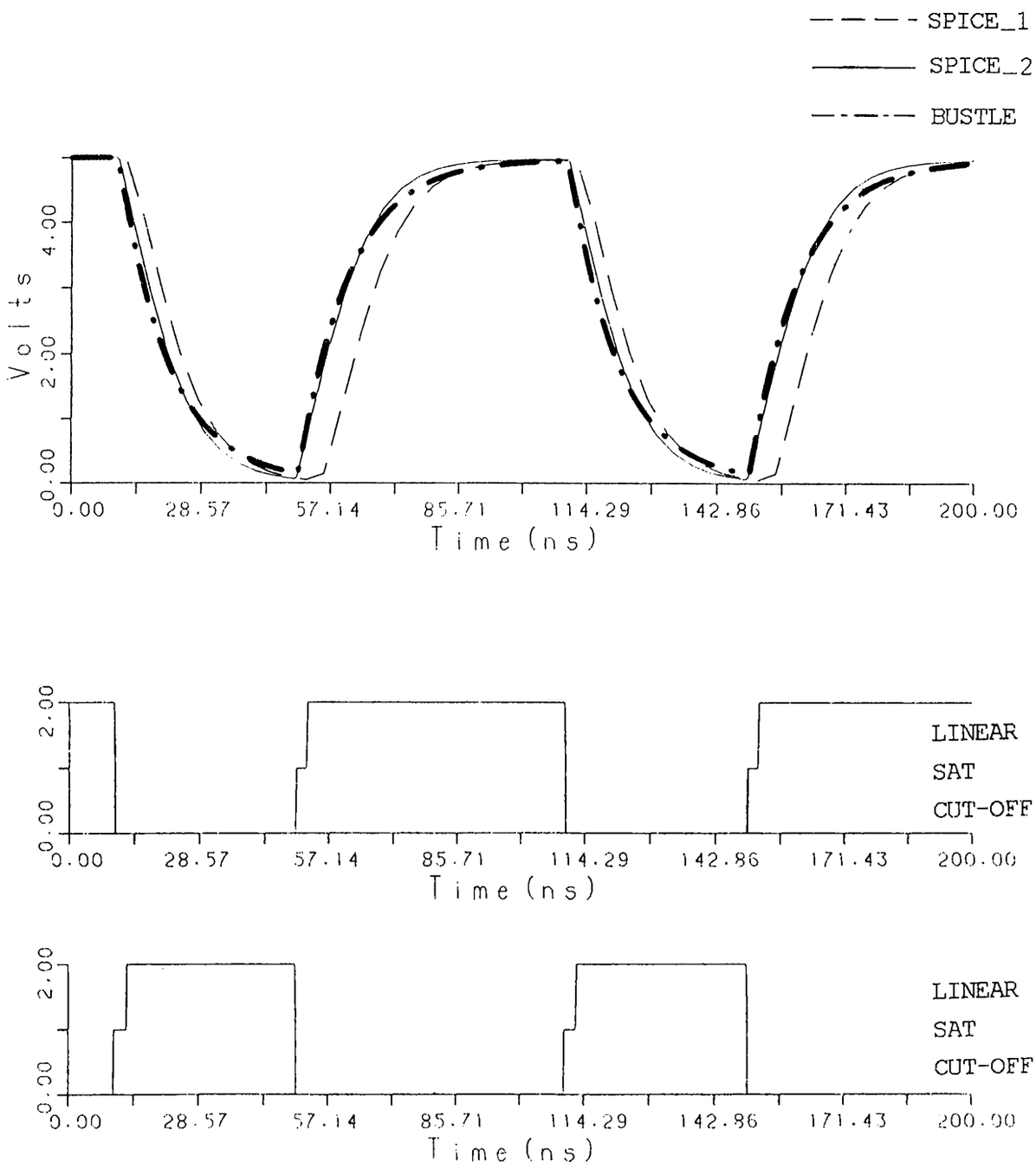

Figure 5.12: The result of transient analysis of the CMOS inverter, and the operating segments of the transistors.

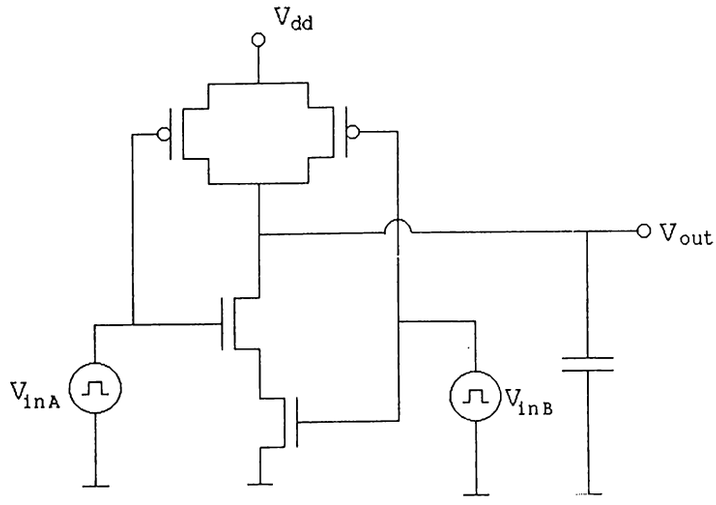

Figure 5.13: The circuit of CMOS NAND Gate.

of the transient analysis are almost the same for both BUSTLE and SPICE. The normalized RMS difference in the results of BUSTLE is 3.87% when compared to SPICE, which is quite small. The CPU time required for the transient analysis of the CMOS NAND gate is 7.5 seconds for BUSTLE, and 36 seconds for SPICE. Consequently, the simulation time for BUSTLE is 1/5 of SPICE with a negligible loss in accuracy. The operating regions of the transistors are also given in Figure 5.15 which helps us a lot to understand the behavior of the circuit. This facility seems to be more useful in the analysis or design of analog circuits containing transistors, (e.g. opamps).

CMOS Logic Function Unit: Figure 5.16.
The last example is a CMOS logic unit which performs the following function.

$$f = \overline{A.B + (C+D).E}$$

The CMOS circuit implementation is done with 10 transistors, and the circuit schematic is given in Figure 5.16. The inputs B,C and D is connected to 5 volts, the input waveform for A and E is shown in Figure 5.17. The output of the unit is loaded by a 1-pf capacitor. The transient analysis is performed using BUSTLE and SPICE and the results of BUSTLE is observed in Figure 5.17. The results seems to be true, that is the circuit is functioning as expected. But we couldn't compare it by SPICE, because it gives an error message

INTERNAL TIMESTEP TOO SMALL IN TRANSIENT ANALYSIS

and aborts from the program. Although, we tried to run SPICE by adjusting the parameters in the option card, it was not able to finish the simulation.

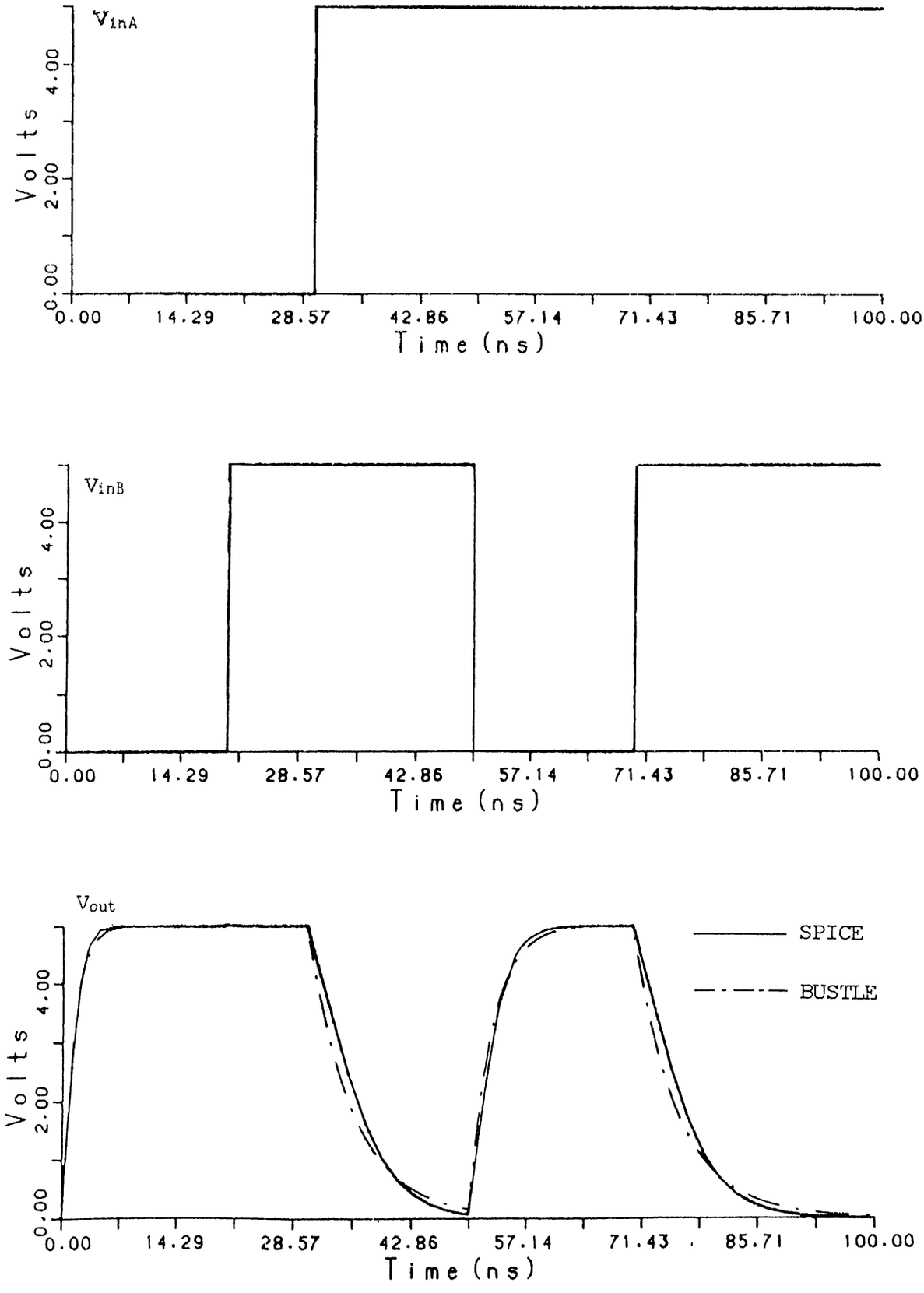

Figure 5.14: The result of the transient analysis of the CMOS NAND Gate.

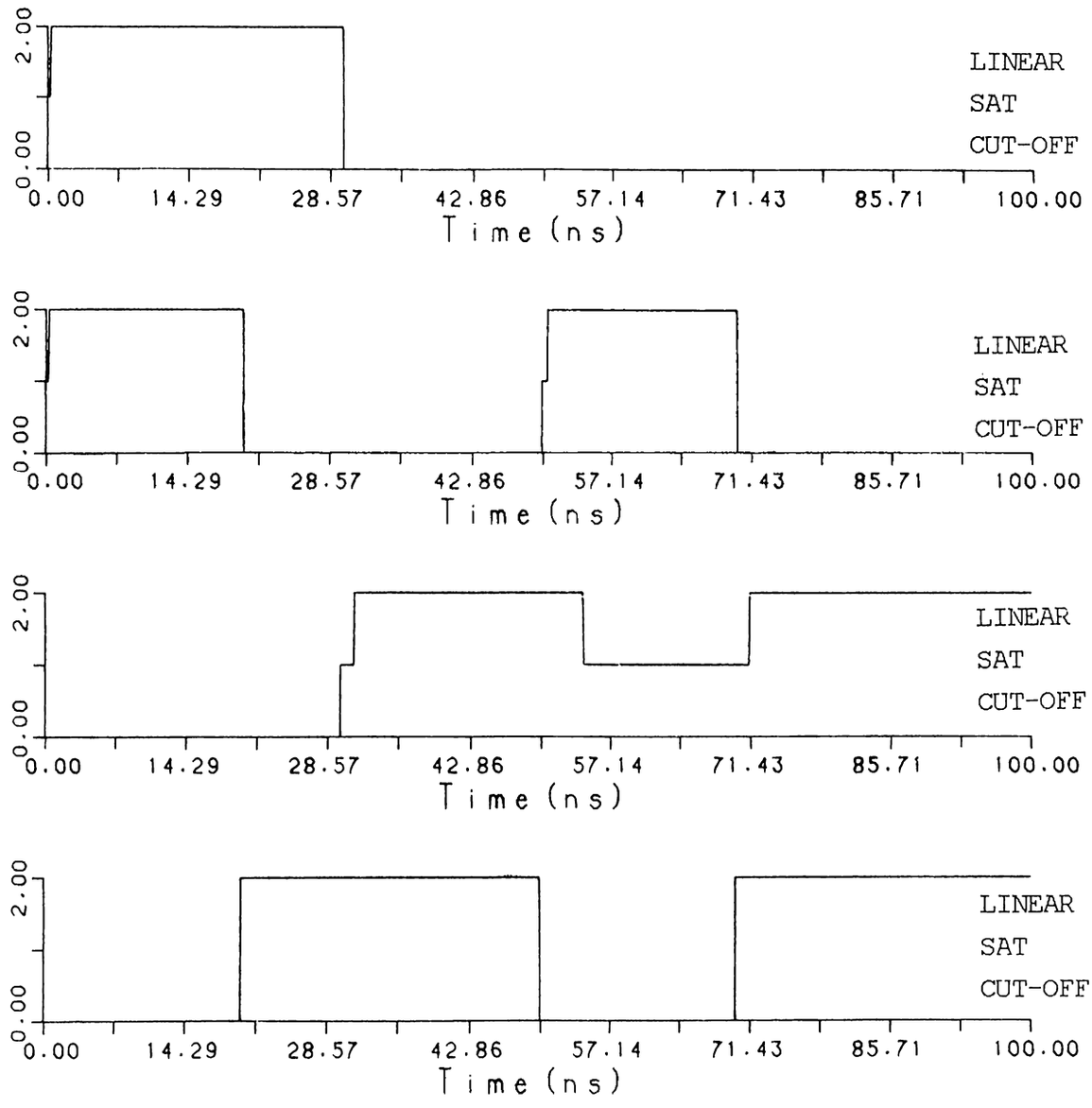

Figure 5.15: Operating segments of the transistors in the CMOS NAND Gate.

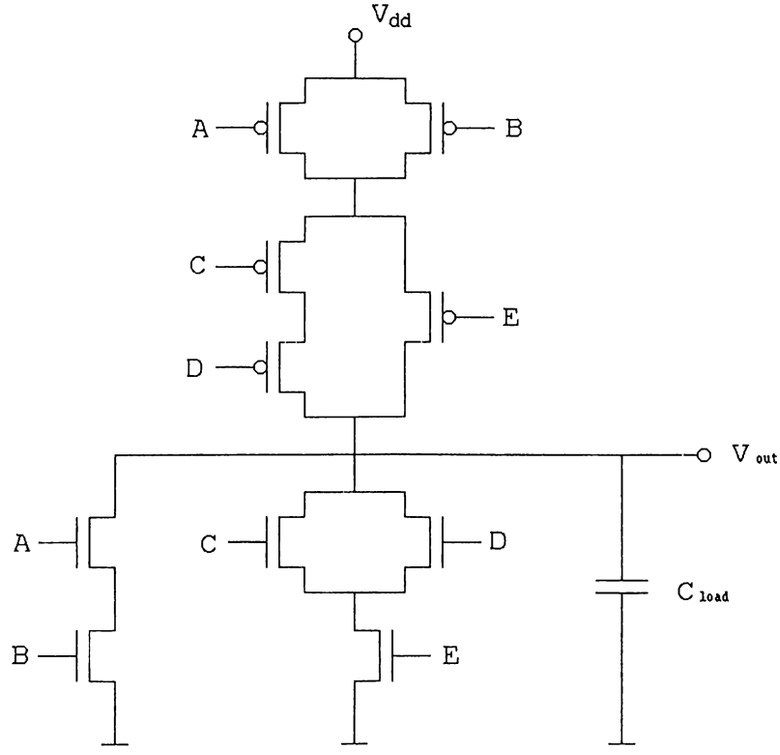

Figure 5.16: The circuit of CMOS Logic Function Unit.

The CPU time required for the BUSTLE simulation is 10.45 seconds.

There are some other interesting examples which demonstrates the efficiency of BUSTLE, they are reported in [1].

As seen from the examples, the results of BUSTLE is quite accurate, and it is faster than SPICE, even in small circuits. Although the program is not optimized for speed yet, it shows a very good performance. It is observed that the performance of BUSTLE increases as the size of the circuit grows larger. BUSTLE can also solve the circuits which can not be simulated using SPICE.

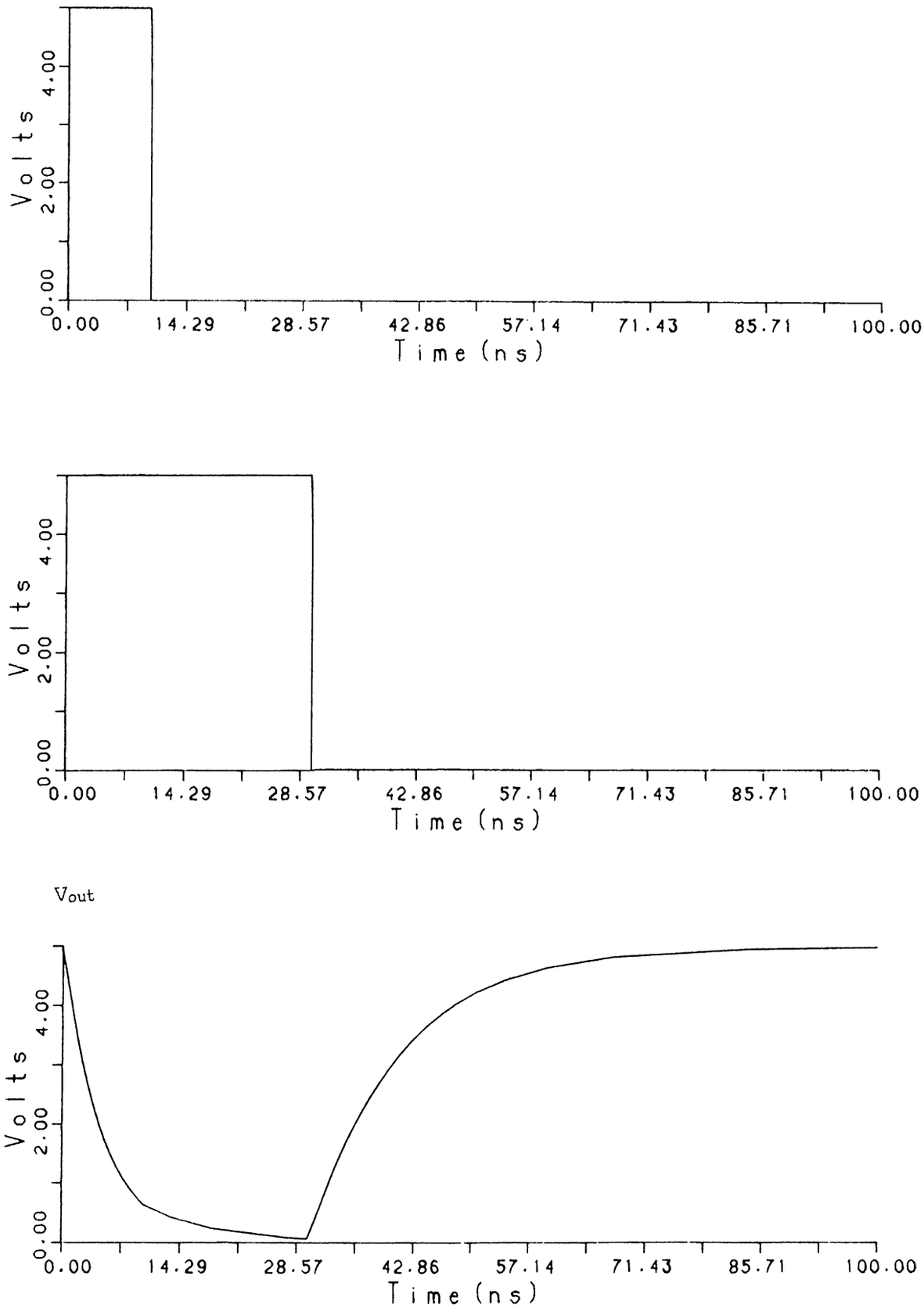

Figure 5.17: Transient response of the CMOS Logic Function Unit.

Chapter 6

CONCLUSION

A new general purpose circuit simulation tool, BUSTLE, using AWE technique and PWL models, is implemented. The results are so promising that this kind of approach will probably dominate the other approaches, especially for large circuits. Some of the results we obtained are:

- Using different combinations of derivative and integral moments for each state variable, we manage to get rid of the instability problem of AWE.
- Using PWL approach, the convergence is guaranteed in DC analysis.
- User defined modeling brings the power to keep pace with the emerging technology and leaves the accuracy speed trade-off to the user.
- Adaptive calculation of time step provides an activity driven feature to the simulator.
- Simple models of nonlinear devices with few segments give quite good results for transient analysis.
- Giving the operating segments of the PWL devices, BUSTLE makes it easier to observe the behavior of the circuit.

The results we have obtained show that BUSTLE may be used in the analysis of digital VLSI circuits effectively. In addition, it can also be used in the analysis and design of analog integrated circuits and provides a lot of help to the user in understanding the circuit behavior.

But there are still some other work that shall need to be done. First, we need to find an algorithm to determine the order of approximation and the

number of derivative moments in the approximation automatically that will check the accuracy, and guarantees the stability. Furthermore, AWE can be directly performed for the nonlinear devices and the outputs which makes the use of different transient time step for each, possible and efficient. Another research topic is to include the effect of transmission lines in the simulation by the direct computation of the moments from the geometry. Also nonlinear capacitors, especially the nonlinear parasitic capacitances of the nonlinear devices need to be considered in the simulation. Efficient PWL modeling of the nonlinear devices and partitioning in LU decomposition are also other research topics that will increase the efficiency of the simulation.

Appendix A

BUSTLE USER'S GUIDE

BUSTLE is a general-purpose circuit simulation program. In order to use it efficiently its input format is made similar to that of existing simulators, i.e. SPICE.

In order to read and analyze the input file, *yacc* (yet another compiler-compiler) and *lex* (lexical analyzer generator) is used.

Lex is a program generator designed for lexical processing of character input streams. It accepts a high-level, problem oriented specification for character string matching, and produces a program in a general-purpose language which recognizes regular expressions. *Lex* source is a table of regular expressions and corresponding program fragments. The table is translated to a program *yylex*. The *yylex* program recognizes expressions in a stream and performs specified actions for each expression as it is detected.

Yacc provides a general tool for imposing structure on the input to a computer program. The *yacc* programmer prepares a specification of the input process; this includes rules describing the input structure, code to be invoked when these rules are recognized, and a low level routine to do the basic input. *yacc* then generates a function to control the input process. This function is called a parser, calls the programmer supplied low level input routine (the lexical analyzer) to pick up the basic items (called *tokens*)from the input stream.

As *lex* can be used with a parser generator to perform lexical analysis phase, it is easy to interface *lex* and *yacc*. *Lex* recognizes only regular expressions; *yacc* writes parsers that accept large class of context-free grammers but requires a lower level analyzer to recognize input tokens.

A.1 INPUT FORMAT

The input format for BUSTLE is of free format type. Fields on a card are separated by one or more blanks. In order to pass from one card to another, a $< RETURN >$ must be entered. A card may be continued by entering a + sign in the beginning of the following card; BUSTLE continues reading after the + sign.

A name field must begin with a letter (A through Z), and cannot contain any delimiters.

A number field may be an integer field, a floating point field, either an integer or floating point number followed by an integer exponent, or either an integer or a floating point number followed by one of the following scale factors.

G=1E9 MEG=1E6 K=1E3 M=1E-3 U=1E-6

N=1E-9 P=1E-12 F=1E-15

A.2 CIRCUIT DESCRIPTION

The circuit to be analyzed is described to BUSTLE by a set of element cards, which define the circuit topology and element values, and a set of control cards, which define the required type of circuit analysis and a set of model cards which define the model parameters.

The first card must be the BEGIN card which initiates the reading of the input, and the last card must be the END card. The other cards must be in the following order:

- BEGIN card
- MODEL cards
- ELEMENT cards
- CONTROL cards
- END card

Each element in the circuit is specified by an element card that contains the element name, the nodes to which the element is connected and the values of

the parameters which determine the electrical characteristics of the element. The first letter of the element name specifies the element type.

Nodes must be nonnegative integers and should be numbered sequentially. The ground node must be numbered zero. The branch numbers are given internally. The circuit can not contain a loop of inductors and a cutset of capacitors. Each node in the circuit must have a DC path to ground.

A.3 BEGIN CARD, COMMENT CARD, END CARD

A.3.1 Begin Card

The input deck must always begin with the begin card.

```
.BEGIN
```

A.3.2 Comment Card

General Form

```
* < any comment >
```

or

```
#< any comment >
```

"*" and "#" in the beginning of a line indicates that this card is a comment card. Comment cards may be placed anywhere in the circuit description.

A.3.3 End Card

The input deck must always end with the end card.

```
.END
```

A.4 ELEMENT CARDS

A.4.1 Resistors

General form:

RXXXXXXX N1 N2 VALUE

Examples:

```
RLOAD 9 0 100
Rsource 1 2 100
r1 12 8 1K
```

N1 and N2 are the two element nodes. VALUE is the resistance (in ohms), and may be positive, negative or zero.

A.4.2 Capacitors and Inductors

General form:

CXXXXXXX N+ N– VALUE < *INCOND* >

LXXXXXXX N+ N– VALUE < *INCOND* >

Examples:

```
Ccc 1 2 3uf 2v
L3 71 20 1mh
```

N+ and N– are the positive and negative nodes of the element, respectively. VALUE is the capacitance in Farads or the inductance in Henries. For the capacitor, the (optional) initial condition is the initial value of capacitor voltage in volts. For the inductor, the (optional) initial condition is the initial value of inductor current in amperes that flows from N+ to N–.

A.4.3 Linear Dependent Sources

BUSTLE allows circuits to contain linear dependent sources characterized by any of the four equations

$$i = gv_c \qquad v = ev_c \qquad i = fi_c \qquad v = hi_c$$

where g, e, f, and h are constants representing transconductance, voltage gain, current gain, and transresistance, respectively.

Linear Voltage-Controlled Current Sources

General form:

GXXXXXXX N+ N– NC+ NC– VALUE

Examples:

```
G5 3 4 7 1 1mmho
```

N+ and N– are the positive and negative nodes respectively. NC+ and NC– are the positive and negative controlling nodes respectively. VALUE is the transconductance (in mhos).

Linear Voltage-Controlled Voltage Sources

General form:

EXXXXXXX N+ N– NC+ NC– VALUE

Examples:

```
eamp1 13 5 3 0 4.2
```

N+ and N– are the positive and negative nodes respectively. NC+ and NC– are the positive and negative controlling nodes respectively. VALUE is the voltage gain.

Linear Current-Controlled Current Sources

General form:

FXXXXXXX N+ N– NC+ NC– VALUE

Examples:

```
F1 15 5 2 7 5
```

N+ and N– are the positive and negative nodes respectively. NC+ and NC– are the node numbers of the controlling branch. VALUE is the current gain.

Linear Current-Controlled Voltage Sources

General form:

HXXXXXXX N+ N– NC+ NC– VALUE

Examples:

```
hv1 33 0 7 0 1.2M
```

N+ and N– are the positive and negative nodes respectively. NC+ and NC– are the node numbers of the controlling branch. VALUE is the transresistance (in ohms).

A.4.4 Independent Sources (Time Invariant)

General form:

VXXXXXXX N+ N– VALUE

IXXXXXXX N+ N– VALUE

Examples:

```
vin 1 2 2v
Is 5 0 4.5mA
```

N+ and N– are the positive and negative nodes respectively. VALUE is the value of the source (in volts or amperes).

A.4.5 Time Varying Independent Sources

Any independent source can be assigned a time-dependent value for transient analysis. There are two independent source functions : pulse, piece-wise linear.

1. **PULSE:** PULSE V1 V2 TD TR TF PW PER

Parameter	Description	Units
V1	initial value	Volts or Amps
V2	pulsed value	Volts or Amps
TD	delay time	seconds
TR	rise time	seconds
TF	fall time	seconds
PW	pulse width	seconds
PER	period	seconds

Examples:

```
VIN 1 0 pulse -1v 1v 2ms 2ms 2ms 40ms 90ms
Vs 3 0 pulse 0v 5v 5ns 0ns 0ns 20ns 50ns
```

2. **PIECE-WISE LINEAR:** PWL T1 V1 <T2 V2 ..... >

Examples:

```
VIN 3 0 pwl 0ms 0v 1ms 1v 2ms 1v 2ms 0v
```

Each pair of values (T_i,V_i) specifies that the value of the source is V_i (in volts or amperes) at time = T_i. The value of the source at intermediate values of time is determined by using linear interpolation on the input values.

Note that it is possible to define an ideal step using time varying independent sources.

A.5 PWL DEVICES

The nonlinear elements of a circuit must be modeled as Piece-wise Linear in order to be used in BUSTLE. Each PWL element card contains the device name, the nodes to which the device is connected and the device model name.

The characteristics of the PWL device is described in a separate model card. More than one element having the same characteristics may use the same model.

A.5.1 Two Terminal PWL Devices

General form:

DXXXXX N+ N– "MNAME"

Examples:

```
dbridge 3 4 "diode"
```

N+ and N– are the positive and negative nodes respectively. MNAME is the model name.

A.5.2 Three Terminal PWL Devices

General form:

TXXXXX N1 NC N2 "MNAME"

Examples:

```
T8 3 5 7 "pnp"
```

N1, NC, N2 are the nodes to which nonlinear device is connected. NC is the common node. MNAME is the model name.

A.6 MODEL CARDS

A.6.1 Two Terminals

General form:

.MODEL2 "MNAME" NOP pt V1 I1 pt V2 I2 < pt V3 I3 ... >

Examples:

```
.MODEL2 "d1" 3 pt -50v -50uA pt 0.6v 0.6uA pt 10v 10A
```

MNAME is the model name. **NOP** is the number of points used in the description. $V_i I_i$ are the sample voltage and current values, respectively that needs to be extracted from the nonlinear characteristics. BUSTLE assumes that the i-v characteristics is linear between these points. Note that, $n+1$ points are needed to define n segments.

A.6.2 Three Terminals

General form:

.**MODEL3** "MNAME" < C_1 C_2 C_3 > **NOR pl** e_{1,v_1} e_{1,i_1} e_{1,v_2} e_{1,i_2} $e_{1,c}$ e_{2,v_1} e_{2,i_1} e_{2,v_2} e_{2,i_2} $e_{2,c}$ **NOB bd NBN nb** b_{v_1} b_{i_1} b_{v_2} b_{i_2} b_c <**bd NBN nb** b_{v_1} ... > <**pl** e_{1,v_1} ... >

Examples:

```
.MODEL3 "nmos" 1pf 0pf 0pf 4
+ pl 0 1 0 0 0 0 0 1 -1e7 0 2
  + bd 1 nb -1 0 0 0 1 bd 3 nb -1 0 1 0 1
+ pl 0 1 0 0 0 -400 0 -1 1e7 400 2
  + bd 0 nb 1 0 0 0 -1 bd 2 nb -1 0 1 0 1
+ pl 0 1 0 0 0 0 0 401 -1e7 0 2
  + bd 3 nb 1 0 0 0 -1 bd 1 nb 1 0 -1 0 -1
+ pl 0 1 0 0 0 400 0 -401 1e7 -400 2
  + bd 2 nb -1 0 0 0 1 bd 0 nb 1 0 -1 0 -1

.MODEL3 "pmos" 1pf 0pf 0pf 4
+ pl 0 1 0 0 0 0 0 1 -1e7 0 2
  + bd 1 nb 1 0 0 0 1 bd 3 nb 1 0 -1 0 1
+ pl 0 1 0 0 0 400 0 1 -1e7 400 2
  + bd 0 nb -1 0 0 0 -1 bd 2 nb 1 0 -1 0 1
+ pl 0 1 0 0 0 0 0 401 -1e7 0 2
  + bd 3 nb -1 0 0 0 -1 bd 1 nb -1 0 1 0 -1
+ pl 0 1 0 0 0 -400 0 401 -1e7 -400 2
  + bd 2 nb 1 0 0 0 1 bd 0 nb -1 0 1 0 -1
```

MNAME is the model name. C_1, C_2, and C_3 are the intrinsic capacitance values between the nodes 1 and common, 2 and common, and 1 and 2 of the PWL element, respectively. If the value of an intrinsic capacitor is given as zero then it is omitted. **NOR** is the number of regions used in the description. **pl** is written to indicate the beginning of a region. Each region is defined by two branch equations and a number of boundary equations. Branch equations are as follows:

$$e_{1,v_1}v_1 + e_{1,i_1}i_1 + e_{1,v_2}v_2 + e_{1,i_2}i_2 + e_{1,c} = 0$$
$$e_{2,v_1}v_1 + e_{2,i_1}i_1 + e_{2,v_2}v_2 + e_{2,i_2}i_2 + e_{2,c} = 0$$

where v_1, i_1, v_2, i_2 are defined in Fig. 2.2.

NOB is the number of the boundaries related to the given region. A boundary is defined by the following equation:

$$b_{v_1}v_1 + b_{i_1}i_1 + b_{v_2}v_2 + b_{i_2}i_2 + b_c \geq 0$$

bd indicates the beginning of a boundary. **NBN** is the ID. number of the region which is the other neighbor of this boundary. ID. number of the region which is passing through the origin (satisfying **0**) is 0, and this region must be given at the first place in the region list. ID. numbers of the other regions are numbered sequentially according to their order in the list.

A.7 CONTROL CARDS

A.7.1 TRAN Card

General form:

.TRAN <TSTEP> TSTOP <TSTART> <UTS>

Examples:

```
.TRAN 1ns 100ns 10ns

.TRAN 1ns 100ns uts

.TRAN 100ns
```

TSTEP is the maximum internal time step that is allowed. TSTART and

TSTOP are the initial time and final time of the transient analysis respectively. If TSTART is omitted, it is assumed to be zero. One can omit TSTEP if TSTART is also omitted. The effect of TSTEP can be removed by assigning it to a large value. If UTS (Use Time Step) is used then the internal time step is directly chosen as TSTEP.

A.7.2 PRINT Card

General form:

.PRINT PRTYPE OV1 <OV2 ... OV8>

Examples:

```
.PRINT TRAN VOUT 8 0 IS 1 0

.PRINT TRAN VIN 1 0 IIN 1 0

.PRINT TRAN V5 5 0 SEG2 12

.PRINT TRAN VIN 2 0 VOUT 16 0 IDD 1 0 SEG3 8 SEG2 7
```

PRTYPE shows whether the output(s) is (are) for a transient or for a DC analysis. The form for voltage, current or segment output variables is as follows:

V N1< N2> specifies the voltage difference between nodes N1 and N2. If N2 is omitted, ground (0) is assumed.

I N1 N2 specifies the current flowing in the branch that is between the nodes N1 and N2. There should be an element between the nodes N1 and N2.

SEG2 IDN specifies the segment number of the two terminal PWL device with the device id. no IDN. The device id. no's are numbered sequentially according to their order in the input deck.

SEG3 IDN specifies the segment number of the three terminal PWL device with the device id. no IDN. The device id. no's are numbered for three terminal PWL devices same as the two terminal elements.

A.7.3 OPTION Card

General form:

.OPTIONS OPT1 OPT2 ... (or OPT=OPTVAL)

Examples:

```
.OPTIONS ORDER=2 REFNUM=1
```

ORDER is the minimum order of approximation in AWE. Default is 1.

NOFDER is the initial number of the derivative moments used in AWE. Default is zero, which means no derivatives is used in the approximation.

REFNUM is the number of the refinements done while solving the circuit. In general there is no need for refinement. Default is zero, which means no refinement.

DEBUG is the level of printing the debugging material in the *infile*.info file. Default is zero, which does not create the *infile*.info file.

TSFP is the Time Step Finding Period. BUSTLE computes the time step in transient analysis dynamically after every TSFP time steps. Default is 1.

SAFETY is the safety factor used in the calculation of the internal time step. The larger the SAFETY, smaller the internal time step. Default is 4.

A.8 EXAMPLE INPUT FILES

14^{th} Order RLC Ladder Circuit:

```
*** AN RLC TREE WITH ORDER 14 ***
.BEGIN
vin 1 0 pwl 0ms 0v 0ms 3v
rs 1 2 10
l1 2 3 100mh 5mA
c1 3 0 10uf 0v
r1 3 4 5
l2 4 5 100mh -1mA
c2 5 0 1uf 0v
r2 5 6 5
l3 6 7 100mh -0.5mA
c3 7 0 1uf 0v
r3 7 8 5
l4 8 9 10mh 1mA
c4 9 0 1uf 0v
r4 9 10 5
l5 10 11 10mh -1mA
c5 11 0 1uf 0v
r5 11 12 5
l6 12 13 10mh 1mA
c6 13 0 1uf 0v
r6 13 14 5
l7 14 15 10mh 0mA
c7 15 0 1uf 0v
r7 15 0 50
.OPTION ORDER=3
.TRAN 0.2ms 20ms uts
.PRINT tran v1 15 0
.END
```

Bridge Rectifier:

```
*** FULL-WAVE RECTIFIER ***
.BEGIN
.MODEL2 "d1" 3 pt -50v -50nA pt 0.6v 0.6nA pt 10v 10A
vin 1 0 pwl 0ms -5v 1ms 5v 2ms -5v 3ms 5v 4ms -5v 5ms 5v
rs 1 2 10
da 2 3 "d1"
db 4 2 "d1"
dc 0 3 "d1"
dd 4 0 "d1"
rl 3 4 1000
.TRAN 0.05ms 5ms 0ms
*** WE WILL OBSERVE THE SEGMENTS OF THE DIODES A & C
.PRINT tran vout 3 4 seg2 0 seg2 2
.END
```

CMOS Inverter:

```
### THE CMOS INVERTER CIRCUIT ###
.BEGIN
*********************************************
.MODEL3 "nmos" 0.05pf 0pf 0pf 4
+ pl 0 1 0 0 0 0 0 1 -1e7 0 2
  + bd 1 nb -1 0 0 0 1 bd 3 nb -1 0 1 0 1
+ pl 0 1 0 0 0 -1800 0 -1 1e7 1800 2
  + bd 0 nb 1 0 0 0 -1 bd 2 nb -1 0 1 0 1
+ pl 0 1 0 0 0 0 0 1801 -1e7 0 2
  + bd 3 nb 1 0 0 0 -1 bd 1 nb 1 0 -1 0 -1
+ pl 0 1 0 0 0 1800 0 -1801 1e7 -1800 2
  + bd 2 nb -1 0 0 0 1 bd 0 nb 1 0 -1 0 -1
#-----------------------------------------#
.MODEL3 "pmos" 0.05pf 0pf 0pf 4
+ pl 0 1 0 0 0 0 0 1 -1e7 0 2
  + bd 1 nb 1 0 0 0 1 bd 3 nb 1 0 -1 0 1
+ pl 0 1 0 0 0 1800 0 1 -1e7 1800 2
  + bd 0 nb -1 0 0 0 -1 bd 2 nb 1 0 -1 0 1
+ pl 0 1 0 0 0 0 0 1801 -1e7 0 2
  + bd 3 nb -1 0 0 0 -1 bd 1 nb -1 0 1 0 -1
+ pl 0 1 0 0 0 -1800 0 1801 -1e7 -1800 2
  + bd 2 nb 1 0 0 0 1 bd 0 nb -1 0 1 0 -1
*********************************************
vin 1 0 pulse 0v 5v 10ns 0ns 0ns 40ns 100ns
vdd 3 0 5v
r1 1 2 1k
r2 3 4 1
tp 2 4 5 "pmos"
tn 2 0 5 "nmos"
cload 5 0 2pf
.TRAN 4ns 200ns uts
.PRINT tran v1 2 0 v2 5 0 seg3 0 seg3 1
.END
```

CMOS FULL ADDER CIRCUIT:

```
*** CMOS FULL ADDER CIRCUIT (28 TRANSISTOR) ***
.begin
*********************************************
.MODEL3 "nmos" 4
+ pl 0 1 0 0 0 0 0 1 -1e7 0 2
  + bd 1 nb -1 0 0 0 1 bd 3 nb -1 0 1 0 1
+ pl 0 1 0 0 0 -1800 0 -1 1e7 1800 2
  + bd 0 nb 1 0 0 0 -1 bd 2 nb -1 0 1 0 1
+ pl 0 1 0 0 0 0 0 1801 -1e7 0 2
  + bd 3 nb 1 0 0 0 -1 bd 1 nb 1 0 -1 0 -1
+ pl 0 1 0 0 0 1800 0 -1801 1e7 -1800 2
  + bd 2 nb -1 0 0 0 1 bd 0 nb 1 0 -1 0 -1
#-------------------------------------------#
.MODEL3 "pmos" 4
+ pl 0 1 0 0 0 0 0 1 -1e7 0 2
  + bd 1 nb 1 0 0 0 1 bd 3 nb 1 0 -1 0 1
+ pl 0 1 0 0 0 1800 0 1 -1e7 1800 2
  + bd 0 nb -1 0 0 0 -1 bd 2 nb 1 0 -1 0 1
+ pl 0 1 0 0 0 0 0 1801 -1e7 0 2
  + bd 3 nb -1 0 0 0 -1 bd 1 nb -1 0 1 0 -1
+ pl 0 1 0 0 0 -1800 0 1801 -1e7 -1800 2
  + bd 2 nb 1 0 0 0 1 bd 0 nb -1 0 1 0 -1
*********************************************
vdd 1 0 5v
va 15 0 pulse 0v 5v 10ns 0ns 0ns 90ns 100ns
vb 16 0 pulse 0v 5v 40ns 0ns 0ns 90ns 100ns
vc 17 0 0v
tp1 15 1 2 "pmos"
tp2 16 2 3 "pmos"
tp3 15 3 4 "pmos"
tp4 16 1 2 "pmos"
tp5 17 2 4 "pmos"
tn1 17 5 4 "nmos"
tn2 15 0 5 "nmos"
tn3 16 0 5 "nmos"
tn4 16 0 6 "nmos"
tn5 15 6 4 "nmos"
tp6 4 1 14 "pmos"
```

```
tn6 4 0 14 "nmos"
tp7 15 1 7 "pmos"
tp8 16 1 7 "pmos"
tp9 17 1 7 "pmos"
tp10 4 7 10 "pmos"
tp11 15 7 8 "pmos"
tp12 16 8 9 "pmos"
tp13 17 9 10 "pmos"
tn7 4 18 10 "nmos"
tn8 15 0 18 "nmos"
tn9 16 0 18 "nmos"
tn10 17 0 18 "nmos"
tn11 17 11 10 "nmos"
tn12 15 12 11 "nmos"
tn13 16 0 12 "nmos"
tp14 10 1 13 "pmos"
tn14 10 0 13 "nmos"
c1 4 0 2pf 5v
c2 10 0 0.5pf 5v
c3 13 0 1pf 0v
c4 14 0 1pf 0v
.tran 100ns
.print tran v1 15 0 v2 16 0 v3 13 0 vout 14 0
.end
```

References

[1] M. Murat Alaybeyi. "BUSTLE: A new circuit simulation tool," M.S. Thesis, Bilkent University, Ankara, Turkey, 1990.

[2] Lawrence T. Pillage , Ronald A. Rohrer. "Asymptotic Waveform Estimation," *IEEE Trans. Computer-Aided Design*, pp. 352-366, April 1990.

[3] Lawrence T. Pillage. "Asymptotic Waveform Evaluation for Timing Analysis," Ph.D. dissertation, Carnegie Mellon University, Pennsylvania, USA, 1989.

[4] X. Huang, V. Raghavan, R.A. Rohrer. "AWEsim : A Program for the Efficient Analysis of Linear(ized) Circuits," *In ICCAD 90* , 1990.

[5] J.Y. Lee, X. Huang, R.A. Rohrer. "Efficient Pole Zero Sensitivity Calculation in AWE," *In ICCAD 90* , 1990.

[6] L.T. Pillage, X. Huang, R.A. Rohrer. "Asymptotic Waveform Evaluation for Circuits containing Floating Nodes," In Proceedings of the 1990 International Symposium on Circuits and Systems, 1990.

[7] T. Fujisawa , E.S. Kuh. "Piecewise-Linear Theory of Nonlinear Networks," *J. Appl. Math.* , Vol. 22 , No. 2, pp. 307-328, March 1972.

[8] M. J. Chien, E.S. Kuh. "Solving nonlinear resistive networks using piecewise-linear analysis and simplical subdivision," *IEEE Trans. on Circuits and Systems,* Vol. 24, pp. 305-317, June 1977.

[9] H.J. Strayer, D.J. Roulston, P.R. Bryant. "DC Solution Speed in Piecewise Linear Network Analysis Programs," *Electronics Letters* , Vol. 22 , No .3, pp. 165-166, January 1986.

[10] V. Zakian. "Simplification of Linear Time-Invariant Systems by Moment Approximants," Int. J. Control, Vol. 18, No. 3, pp. 455-460, 1973.

[11] M.J. Bosley, H.W. Kropholler, F.P. Lees. "On the Relation Between the Continued Fraction Expansion and Moments Matching Methods of Model Reduction," Int. J. Control, Vol. 18, No. 3, pp. 461-474, 1973.

[12] Y. Shamash. "Stable Reduced-Order Models Using Padeé-Type Approximations," *IEEE Trans. Automatic Control,* pp. 615-616, October 1974.

[13] F.J. Alexandro, JR. "Stable Partial Padeé Approximations for Reduced-Order Transfer Functions." *IEEE Trans. Automatic Control,* Vol. AC-29 No. 2, pp. 159-161, February 1984.

[14] Hu Xiheng. "FF-Padeé Method of Model Reduction in Frequency Domain," *IEEE Trans. Automatic Control,* Vol. AC-32, No. 3, pp. 243-246, March 1987.

[15] C.F. Chen, L.S. Shieh. "A Novel Approach to Linear Model Simplification," Int. J. Control, Vol. 8, No. 6, pp. 561-570, 1968.

[16] Y. Shamash. "Model Reduction Using the Routh Stability Criterion and the Padeé Approximation Technique," Int. J. Control, Vol. 21, No. 3, pp. 475-484, 1975.

[17] M.F. Hutton, B. Friedland. "Routh Approximations for Reducing Order of Linear, Time-Invariant Systems," *IEEE Trans. Automatic Conirol,* Vol. AC-20, No. 3, pp.329-337, June 1975.

[18] J. Pal. "Stable Reduced-Order Padé Approximants Using the Routh-Hurwitz Array," *Electronics Letters* , Vol. 15 , No. 8, pp. 225-226, Aprii 1979.

[19] P.R. Graves-Morris. "The numerical calculation of Padé approximants," In L. Wuytack, editor, *Padé Approximation and its Applications,* Vol. 765 of *Lecture Notes in Mathematics,* pp. 231-245. Springer-Verlag, 1979.

[20] R.A. Rohrer. *Circuit Theory : An Introduction to the State Variable Approach.* McGraw-Hill, Inc., 1970.

[21] Ö. Hüseyin Oral, E.Sezer. *Elementary Matrix Algebra and Differential Equations.* METU, Ankara, 1980.

[22] C.T. Chen. *Linear System Theory and Design.* CBS College Publishing, 1984.

[23] L.O. Chua, P.M. Lin. *Computer-Aided Analysis of Electronic Circuits.* Prentice-Hall, Englewood Cliffs, 1975.

[24] J. Gilbert, L. Gilbert. *Elements of Modern Algebra.* PWS-KENT Publishing Company, Boston, 1988.

[25] P.R. Adby. *Applied Circuit Theory: Matrix and Computer Methods.* John Wiley & Sons, NewYork, 1980.

[26] J. Vlach, K. Singhal. *Computer Methods for Circuit Analysis and Design.* Van Nostrand Reinhold Company, NewYork, 1983.

[27] W.H. Press, B.P. Plannery, S.A. Teukolsky, W.T. Vetterling. *Numerical Recipes in C.* Cambridge University Press, Cambridge, 1988.

[28] N. Weste, K. Eshraghian. *Principles of CMOS VLSI Design, A Systems Perspective.* Addison-Wesley Publishing Company, USA, 1985.

[29] C.A. Desoer, E.S Kuh. *Basic Circuit Theory.* McGraw-Hill, NewYork, 1969.

[30] L.O. Chua, C.A. Desoer, E.S Kuh. *Linear and Nonlinear Circuits.* McGraw-Hill, NewYork, 1987.

[31] Ronald A. Rohrer (private communication).